Red Hat Linux
Herramientas para
la administración de redes

Red Hat Linux
Herramientas para la administración de redes

Steve Maxwell

Traducción

Gustavo Elías Fonseca Bueno

Máster en economía

Universidad de Tsukuba, Japón

Revisión técnica

Jamir Antonio Ávila Mojica

Ingeniero de sitemas

Universidad Nacional de Colombia

Bogotá • Buenos Aires • Caracas • Guatemala • Lisboa • Madrid • México
Nueva York • Panamá • San Juan • Santiago de Chile • São Paulo
Auckland • Hamburgo • Londres • Milán • Montreal • Nueva Delhi • París
San Francisco • San Luis • Sidney • Singapur • Tokio • Toronto

Red Hat Linux Herramientas para la administración de redes

No está permitida la reproducción total o parcial de este libro, ni su tratamiento informático, ni la transmisión de ninguna forma o por cualquier medio, ya sea electrónico, mecánico, por fotocopia, por registro u otros métodos sin el permiso previo y por escrito de los titulares del Copyright.

DERECHOS RESERVADOS. Copyright © 2001, respecto a la primera edición en español, por McGRAW-HILL INTERAMERICANA, S. A. Avenida de las Américas, No. 46-41. Bogotá, D.C., Colombia.

Translated from the first edition of
RED HAT LINUX NETWORK MANAGEMENT TOOLS
Copyright © MM, by McGraw-Hill Companies Inc.
All rights reserved.
P/N: 0-07-212260-9
Part of ISBN: 0-07-212262-5

Editor: Eder Mauricio Hernández B.
Jefe de Producción: Consuelo E. Ruiz M.

1234567890

2034567891

ISBN: 958-41-0220-6

Impreso en Colombia

Printed in Colombia

Se imprimieron 6.500 ejemplares en el mes de mayo de 2001
Impreso por Editorial Nomos S. A.

Para Ginny, por siempre y para siempre...

Contenido

Lista de figuras xi
Lista de tablas xvii
Listados de códigos xxi
Prefacio xxiii

01 Vista general de la administración de redes 1
Componentes de la administración de redes 4

02 Protocolo simple de administración de red 21
Operación del SNMP 23
Base de información de administración (MIB) 25
Vista general de las versiones de SNMP 41
Agente maestro y subagente 55

03 Conjunto de protocolos TCP/IP 59
Conjunto de protocolos TCP/IP 60
Modelo OSI 62
Arquitectura del protocolo TCP/IP 63
Herramientas de usuario final 66
Protocolo de resolución de dirección 99

04 Herramientas y utilidades centrales del sistema 109
Herramienta ARP 110
Herramienta ifconfig 119
Herramienta netstat 129
Herramienta ping 147
Herramienta tcpdump 158
Herramienta traceroute 177

05 Utilidades y herramientas adicionales del sistema 189
Herramienta arpwatch 190
Herramienta ethereal 194

Herramienta fping 216
Herramienta nmap 224
Herramienta xtraceroute 246

06 Vista general de MIB-II 253

07 Uso de los agentes SNMP 331

Vista general de los agentes 332
Consideraciones de seguridad 337
El agente Linux SNMP 338
Agente maestro de Sun 382
El agente SNMP de Sun 396

08 Herramientas SNMP 417

Funciones de seguimiento y administración 418
Comandos UCD 420
Herramienta snmpdelta 429
Herramienta snmpget 431
Herramienta snmpgetnext 433
Herramienta snmpnetstat 435
Herramienta snmpset 440
Herramienta snmpstatus 444
Herramienta snmptable 447
Herramienta snmptest 450
Herramienta snmptranslate 450
Herramienta snmptrap 452
Herramienta snmptrapd 453
Herramienta snmpwalk 457
Herramienta snmpbulkwalk 459
Herramienta snmpconf 462
Herramienta tkmib 466

09 Herramientas habilitadas para la Web 473

Herramienta MRTG 474
Resolución de problemas MRTG 508
Herramienta ntop 510

10 Panel de control Linux 545

El panel de control 546
Uso del configurador de la red 549

11 Herramienta tkined – editor de red 581

Vista general de tkined 582
Administración de la red 604
Referencia de menús 627

A Un vistazo a las herramientas 635

B Procedimientos de instalación y notas sobre el software 639

Uso de RPM para instalar los paquetes de software 642
Procedimientos de instalación de paquetes que no son RPM 648

C Glosario para el trabajo en red/expresiones o términos sobre la administración de red 659

Índice 669

Lista de figuras

1.1	Tres elementos de la administración de red.	2
1.2	Relación administrador/agente.	5
1.3	Agente dentro de un dispositivo de red.	5
1.4	Diagrama funcional de agente.	8
1.5	Agente SNMP y objetos MIB.	9
1.6	Sistema basado en *proxy*.	11
2.1	Relación administrador y agente SNMP.	24
2.2	Estructura y organización MIB.	26
2.3	Explorador MIB.	27
2.4	Ejemplo de estructura MIB.	32
2.5	Estructura del grupo de interface MIB-II.	35
2.6	MIB empresariales de una rama privada.	39
2.7	Formato de mensaje SNMP.	44
2.8	Formato de mensaje GetRequest.	45
2.9	Formato de mensaje GetResponse.	48
2.10	Formato de mensaje Trap.	49
2.11	Arquitectura agente maestro-subagente.	57
3.1	TCP/IP sobre el modelo OSI de trabajo en red.	64
3.2	Grupo de protocolos TCP/IP.	66
3.3	Números de segmento TCP en acción.	74
3.4	Operación de INETD con solicitud Telnet.	76
3.5	Dos sesiones TCP de la misma fuente/destino.	77
3.6	Apertura de una conexión TCP mediante el uso de un acuerdo de conexión de tres vías.	78
3.7	Cierre de una conexión TCP.	79
3.8	Máquina de estado TCP.	80
3.9	Formato del segmento TCP.	81
3.10	Las tres primeras clases principales de dirección IP.	86
3.11	Direccionamiento de las subredes.	88
3.12	Cálculo de la subred.	89
3.13	Subred de muestra.	89
3.14	Ventana principal del cálculo de la subred.	92
3.15	Ventana principal de cálculo de subred.	94

3.16 Solicitud/respuesta de eco ICMP. 95
3.17 Desviación del retiro de fuente ICMP. 97
3.18 Desviación ICMP. 98
3.19 Solicitud ARP. 100
3.20 Respuesta ARP. 101
3.21 Formato del mensaje ARP/RARP. 102
3.22 Formato de la dirección de enlace de datos. 104
3.23 Página de búsqueda de OUI de IEEE. 106
3.24 Resultados de búsqueda de OUI. 107
4.1 Problema de nodo fuera de alcance. 151
4.2 Diagrama del filtro de paquete **tcpdump**. 169
4.3 **traceroute** con un salto. 180
4.4 **traceroute** con dos saltos. 181
5.1 Ventana principal **ethereal**. 195
5.2 Ventana de Preferencias de Captura **ethereal**. 199
5.3 Ventana de Captura/Playback **ethereal**. 199
5.4 Paquete ARP seleccionado, en detalle. 200
5.5 Ventana de filtro de paquete TCP. 203
5.6 Ventana de filtro **ethereal**. 204
5.7 Selección del filtro de difusión IP. 205
5.8 Filtro **ethereal** presentado. 205
5.9 Lista extensa de paquetes clasificados por tipo de protocolo. 206
5.10 Filtro de presentación FTP. 206
5.11 Ventana guardar **ethereal**. 208
5.12 Ventana de preferencias **ethereal**. 209
5.13 Preferencias de impresión **ethereal**. 210
5.14 Avance de la salida **a2ps**. 211
5.15 Ventana de opciones de presentación del campo time **ethereal**. 211
5.16 Paquetes mostrados con la opción Time of Day. 212
5.17 Información de inicio de sesión TCP. 213
5.18 Listado de directorio TCP. 214
5.19 Ventana de resumen del paquete **ethereal**. 215
5.20 Ventana de filtro de paquete TCP. 247
5.21 Ventana del nuevo trazado. 248
5.22 Cuadro de diálogo de registro de nuevo *host* (*new host record*). 248
5.23 Información de salto de destino. 249
6.1 Vista del grupo `mib 2`. 255
6.2 Vista del grupo `system`. 256

6.3 Vista del grupo `interfaces`. 263
6.4 Vista del grupo `at`. 275
6.5 Vista del grupo `ip`. 278
6.6 Vista de la tabla `ipAddrTable`. 284
6.7 Vista de `ipRouteTable`. 286
6.8 Vista de la tabla `ipNetToMediaTable`. 292
6.9 Vista de `ipForward`. 294
6.10 Vista del grupo `icmp`. 296
6.11 Vista del grupo `tcp`. 303
6.12 Vista de la tabla `tcpConnTable`. 307
6.13 Vista del grupo `udp`. 311
6.14 Vista del grupo de transmisión `dot3`. 314
6.15 Vista de la tabla `dot3StateTable`. 315
6.16 Vista del grupo `snmp`. 322
7.1 Ramas Sun/UCD del agente MIB. 337
7.2 Función `pass` del agente. 356
7.3 Vista de árbol de `procTable`. 371
7.4 Vista de `extTable`. 375
7.5 Vista de árbol de `diskTable`. 377
7.6 Vista de árbol de `loadTable`. 380
7.7 Vista jerárquica de `sunMasterAgent`. 390
7.8 Vista de árbol de `sunSubAgentEntry`. 392
7.9 MIB Sun enterprise. 406
7.10 Vista de árbol `sunSystem`. 406
7.11 Vista de la `sunProcessTable`. 409
7.12 Transformación del estado del proceso MIB. 409
7.13 Vista `sunHostPerf`. 414
8.1 Ejemplo del "conjunto de no retorno". 443
8.2 Salida del comando `snmpstatus`. 446
8.3 Captura del paquete de operación `snmpbulkwalk`. 461
8.4 Ventana principal **tkmib**. 467
8.5 Presentación del botón *table*. 468
8.6 Ventana de diálogo de búsqueda de nodo. 468
8.7 Ventana de diálogo de carga del archivo MIB. 469
8.8 Vista de árbol para la rama Cisco. 470
8.9 Cuadro de diálogo número de puerto. 470
8.10 Cuadro de diálogo de la cadena de comunidad. 475
9.1 Ejemplo de una red de oficina.

9.2 Operación básica del MRTG. 476
9.3 Reporte Web MRTG. 477
9.4 Reportes MRTG semanales, mensuales y anuales. 480
9.5 Índice maestro para los dispositivos a los que se les ha hecho seguimiento. 499
9.6 Reporte MRTG del espacio del disco. 503
9.7 Reporte MRTG de uso del disco, en porcentajes. 505
9.8 Página Web principal **ntop**. 513
9.9 Reporte de todos los protocolos. 515
9.10 Reporte IP. 516
9.11 Reporte de rendimiento. 517
9.12 Reporte de flujos de red. 518
9.13 Reporte de todos los protocolos. 520
9.14 Reporte IP. 520
9.15 Reporte de rendimiento. 521
9.16 Reporte de estadísticas de multidifusión. 523
9.17 Primer reporte de estadísticas de tráfico. 525
9.18 Segundo reporte de estadísticas de tráfico. 529
9.19 Reporte de *host*. 529
9.20 Reporte de *host* individual. 530
9.21 Reporte adicional de información del *host*. 532
9.22 Reporte de rendimiento. 532
9.23 Reporte IP remoto/local. 534
9.24 Reporte de matriz de tráfico. 536
9.25 Reporte del uso local (parte uno). 537
9.26 Reporte del uso local (parte dos). 538
9.27 Reporte de distribución de protocolo. 539
9.28 Reporte de subred del uso del protocolo . 540
9.29 Ventana Ntop de inicio de sesión. 542
9.30 Presentación del modo terminal. 544
9.31 Modo terminal de ayuda. 544
10.1 Acceso al panel de control. 547
10.2 Ventana principal del panel de control. 548
10.3 Ventana del configurador de red. 550
10.4 `Configurador de red`, vista de *hosts*. 552
10.5 Ventana de diálogo de *host* agregado. 553
10.6 Configurador de red, vista de la interfaz. 555
10.7 Ventana de selección de la interfaz. 555
10.8 Ventana de interfaz Ethernet. 556

10.9 Ventana de diálogo *Save*. 557
10.10 Ventana de diálogo de interfaz PPP. 558
10.11 Ventana de personalización PPP. 559
10.12 Ventana de marbete Comunicación. 561
10.13 Panel de trabajo en red. 562
10.14 Panel de configuración PAP. 563
10.15 Ventana de clon. 564
10.16 Ventana de alias de interfaz. 564
10.17 Ventana de verificación de remoción de la interfaz. 565
10.18 Panel de enrutado. 566
10.19 Entrada de ruta estática. 567
10.20 Ventana de configuración Linux. 568
11.1 Ventana principal **tkined**. 583
11.2 Ventana de solicitud de descubrimiento de red. 584
11.3 Ventana de estado de descubrimiento. 585
11.4 Topología descubierta de la red. 586
11.5 Lista de iconos de dispositivo. 587
11.6 Vista de topología completada. 587
11.7 Vista del panel de herramientas. 589
11.8 Ejemplo de vista que contiene grupos de objetos. 592
11.9 Grupo expandido. 592
11.10 Objetos de referencia. 593
11.11 Vista referenciada. 593
11.12 Ejemplo de la topología física. 595
11.13 Ejemplo de la topología lógica. 595
11.14 Ventana de diálogo Guardar como. 597
11.15 Ventana de diálogo atributos de los objetos. 598
11.16 Vista lógica. 598
11.17 Cuadro de diálogo de confirmación. 599
11.18 Cuadro de diálogo de atributo del objeto. 600
11.19 Vista del grupo. 601
11.20 Vista LAN completada. 602
11.21 Ventana de diálogo de los atributos del objeto referencia. 603
11.22 Vista de topología WAN. 603
11.23 Menú `Tools`. 604
11.24 Adición de un nuevo menú en el menú principal. 605
11.25 Submenú `IP Trouble`. 606
11.26 Ventana de diálogo de Multi Ping. 607

11.27 Ventana de resultado Ping. 608
11.28 Listado de los servicios RPC. 609
11.29 Ventana de diálogo de intervalo. 610
11.30 Ventana de resultado remoto. 610
11.31 Ventana de parámetro IP-Trouble. 611
11.32 Menú IP-Monitoring. 612
11.33 Cambio de un icono de dispositivo para denotar una falla. 612
11.34 Fallas posibles de seguimiento. 613
11.35 Vista de rendimiento específico. 615
11.36 Vista de rendimiento de `monet`. 616
11.37 Ventana de selección `Modify Job`. 617
11.38 Ventana de la tarea de seguimiento `Modify`. 617
11.39 Menú `IP Layout`. 618
11.40 Menú `IP-Discover`. 621
11.41 Lista de consulta de servidor RPC. 622
11.42 Menú `SNMP-Trouble`. 624
11.43 Ventana de presentación de la información del sistema. 624
11.44 Ventana de opciones de parámetros SNMP. 625
11.45 Menú `SNMP-Monitor`. 626
11.46 Ventana de diálogo de seguimiento. 627
11.47 Gráfico reducido del objeto `ifInOctects.4`. 627
11.48 Desprendimiento del menú `Help`. 628
11.49 Menú `File`. 629
11.50 Menú `Select`. 630
11.51 Diálogo de `Selection by type`. 630
11.52 Selección de tipo. 631
11.53 Menú `Edit`. 631
11.54 Ventana de creación de un atributo nuevo. 632
11.55 Menú `Structure`. 632
11.56 Menú `Icon`. 633
11.57 Menú `Options`. 633
B.1 Ventana principal GNOME RPM. 644
B.2 Ventana de instalación. 645
B.3 Ventana `Add Packages`. 645
B.4 Ventana de instalación con el paquete seleccionado. 646
B.5 Ventana de consulta del paquete. 646
B.6 Ventana de progreso. 647
B.7 Ventana de diálogo del mensaje de advertencia. 647

Lista de tablas

1.1 Lista de MIB estándar. 10
1.2 Lista de MIB de propiedad exclusiva o propietaria. 10
1.3 Servicios del administrador enumerados por categoría. 12
2.1 Tipos de datos universales. 28
2.2 Tipos de datos de aplicación. 29
2.3 Tabla de instancias del grupo interface. 35
2.4 Números asignados a empresas. 39
2.5 Trampas predefinidas de SNMP. 50
2.6 Códigos de respuesta SNMP. 52
3.1 Soporte de protocolo TCP/IP para administración. 61
3.2 Protocolos y servicios centrales TCP/IP. 67
3.3 Servicios adicionales TCP/IP. 72
3.4 Puertos TCP comunes. 75
3.5 Tipos de segmento TCP. 83
3.6 Tipos de mensaje de error ICMP. 94
3.7 Muestra de códigos OUI de vendedor. 105
4.1 Opciones importantes de línea de comandos **arp**. 111
4.2 Indicadores de entrada ARP. 112
4.3 Opciones y palabras clave **ifconfig**. 120
4.4 Direcciones estándar de máscara de red IP. 126
4.5 Palabras clave **netstat**. 129
4.6 Opciones del comando **netstat**. 130
4.7 Estados TCP mostrados con **netstat**. 133
4.8 Objetos MIB del agente Linux que corresponden a la salida **netstat**. 140
4.9 Opciones de comando **ping**. 151
4.10 Opciones de modo de operación **tcpdump**. 161
4.11 Opciones de presentación TCPDUMP. 165
4.12 Primitivas relacionadas con la dirección. 170

4.13	Primitivas de protocolo.	173
4.14	Operadores aritméticos y booleanos.	174
4.15	Ejemplos del comando **tcpdump**.	178
4.16	Códigos de presentación de **traceroute**.	182
4.17	Opciones operacionales **traceroute**.	185
5.1	Opciones de línea de comandos **arpwatch**.	191
5.2	Opciones de presentación **ethereal**.	196
5.3	Opciones funcionales **ethereal**.	196
5.4	Tamaños de marcos de enlace de datos.	198
5.5	Lista de opciones de presentación **fping**.	217
5.6	Lista de opciones operacionales **fping**.	222
5.7	Técnicas de exploración **nmap**.	228
5.8	Opciones **nmap** adicionales.	229
5.9	Listados estándar **nmap** de red.	240
6.1	Lista de grupos MIB-II.	254
6.2	Capas funcionales utilizadas para determinar `syServices`.	257
6.3	Tipos y descripción de interfeces comunes.	266
6.4	Valores **IpRouteProto** definidos.	287
7.1	Vista general funcional de los agentes.	334
7.2	Comandos del agente UCD ordenados por categoría.	340
7.3	Tipos de objetos MIB de agente soportados.	357
7.4	Opciones de línea de comandos del agente maestro.	385
7.5	Opciones de comandos del agente maestro.	397
7.6	Opciones de configuración del agente.	398
7.7	Comparación entre los objetos MIB-II y los objetos Sun MIB.	403
8.1	Herramientas UCD SNMP.	421
8.2	Opciones comunes de presentación.	423
8.3	Opciones operacionales comunes.	426
8.4	Opciones de línea de comandos adicionales `snmpdelta`.	430
8.5	Opciones de línea de comandos `snmpnetstat`.	436
8.6	Tipos de objeto `snmpset`.	442
8.7	Opciones adicionales `snmptable` de la línea de comandos.	448

8.8 Opciones de línea de comandos **snmptranslate**. 451
9.1 Bytes por segundo para interfaces diferentes. 485
9.2 Comandos globales opcionales. 488
9.3 Opciones de personalización MRTG. 490
9.4 Opciones de conmutadores booleanos. 493
9.5 Argumentos de línea de comandos **indexmaker**. 497
9.6 Objetos predefinidos `MRTG` SNMP. 500
9.7 Opciones de línea de comandos de la secuencia de comandos **df2mrtg**. 504
9.8 Módulos adicionales para el MRTG. 506
9.9 Umbrales de rendimiento del reporte de matriz de tráfico. 535
9.10 Opciones de línea de comandos. 541
10.1 Funciones disponibles de la categoría del trabajo en red. 570
10.2 Funciones disponibles de la categoría cuenta de usuario. 574
10.3 Funciones adicionales de la categoría **config**. 576
10.4 Funciones disponibles bajo la categoría `Control`. 578
11.1 Submenús del menú de seguimiento IP. 606
A.1 Lista de herramientas descritas en este libro. 636
B.1 Directorio de información de las herramientas. 640
C.1 Expresiones y significados. 660

Listados de códigos

4.1	Secuencia de comandos de inicio ARP.	119
4.2	Resultados de **netstat --help**.	131
5.1	Secuencia de comandos de ejemplo de **fping** para identificar los sistemas abajo.	219
5.2	Base de datos personal del usuario para **xtraceroute**.	251
7.1	Secuencia de comandos `passtest`.	359
7.2	Secuencia de comandos **Perl** (`maintain messages.pl`). para mantener el archivo de mensajes.	362
7.3	Un "recorrido" SNMP del archivo `/var/log/messages`.	366
7.4	Ejemplo de configuración del agente.	368
7.5	Archivo de registro del agente.	383
7.6	Archivo de recurso del agente.	384
7.7	Ejemplo de un archivo de configuración del agente.	401
8.1	Ejemplo de la tabla UDP.	434
8.2	Secuencia de comandos de inicio UCD con el comando **snmptrapd**.	455
8.3	Archivo `syslog.conf` estándar con entrada **trap**.	456
8.4	Ejemplo del archivo `SNMPCONF`.	463
9.1	Ejemplo del archivo de configuración **cfgmaker**.	481
9.2	Entrada de interfaz comentada.	482
B.1	Secuencia de comandos de inicio del agente.	652
B.2	Arranque del agente UCD.	656

Prefacio

Este libro analiza los fundamentos de la administración de redes desde la perspectiva de los componentes individuales. Muchas herramientas de software que proporcionan importantes funciones y características que se encontraban previamente sólo en productos y soluciones comerciales están disponibles ahora para el público. Aquí se estudian y describen muchas de las herramientas y utilidades discretas que se pueden usar para manejar de manera efectiva una red empresarial, que consta de estaciones de trabajo, servidores y otros componentes de red, como conmutadores, enrutadores y concentradores (*hubs*).

Durante los últimos años, las redes empresariales han llegado a ser el enlace crítico y el componente integral del campo de sistemas de información. Lejos están los días en que la conectividad entre los usuarios era opcional y las fallas en la red eran algo común en la computación. En muchas corporaciones e instituciones, un apagón o una demora en la red puede tener serias implicaciones en la capacidad de la organización para realizar sus actividades normales de negocios o para comunicarse con los clientes. En la comunidad financiera, un apagón, aun relativamente corto, en la red puede generar un impacto financiero significativo.

En la actualidad, las redes corporativas e institucionales se caracterizan por un crecimiento significativo en la diversidad y el número de sistemas conectados con estas redes. Esto se conoce como *trabajo heterogéneo en red* y es bastante común. Es difícil el manejo efectivo de muchas y diferentes computadoras, además de periféricos y dispositivos centrales, porque la mayor parte de este manejo debe hacerse de forma manual. Una falla crítica del sistema o de la red puede tener un impacto significativo en el uso de los servicios corporativos y afectar las operaciones cotidianas de una organización. Se han establecido también muchas redes,

con muy poca previsión respecto a su operatividad o sus capacidades de actualización. Esto dificulta el trabajo de administración de redes, debido a los requerimientos adicionales de sistemas heredados.

Además, con la aceptación generalizada de la Internet como el vehículo principal de comunicación externa, para muchas compañías es aún más crítico mantener la red actualizada, porque los usuarios y los clientes están acostumbrados al acceso instantáneo a información y servicios. Con los pocos pasos involucrados en el acceso a un sitio Web desde cualquier lugar del mundo, una desconexión de Internet podría significar la pérdida de oportunidades de ingresos, si los clientes se deciden por el sitio de un competidor.

Debido al crecimiento exponencial en la industria de redes, el campo de la computación está lleno de varias plataformas de computación y sistemas que ofrecen muchos y variados distribuidores. Gracias al conjunto (*suite*) de protocolos TCP/IP, muchas clases de sistemas y dispositivos de computación se pueden conectar a la red, en un rango que va desde grandes sistemas *mainframe*, hasta pequeños dispositivos manuales. Con TCP/IP, muchos de estos sistemas comparten servicios comunes y suministran un marco de trabajo estándar para conectividad. Además, con la implementación de protocolos para la administración de redes entre estos sistemas, se puede establecer un modelo estándar de administración de redes.

Este libro proporciona el conocimiento de herramientas críticas y las habilidades necesarias para la administración de sistemas y redes empresariales. La importancia fundamental y la meta de este libro es proporcionar información sobre el uso de herramientas de software y utilidades del sistema relacionadas con la administración de una empresa TCP/IP.

Audiencia

Este libro está dirigido fundamentalmente a los administradores de redes y a los administradores de sistemas. Si ya conoce la diferencia entre los comandos **arp** e **ifconfig** y ha usado el comando **snmpwalk**, este libro puede no ser para usted. Otros, que no están familiarizados con el uso de estos comandos, encontrarán útil el libro ya que trata los fundamentos de la administración de redes, los protocolos específicos de administración del trabajo en red y las herramientas de software ya mencionadas. Les servirá

como punto de partida a aquellos que tienen que ver con el soporte de redes en el mundo real. El texto se enfoca más hacia herramientas que hacia descripciones largas y detalladas de la operación de los protocolos, el diseño de administración de redes, o su arquitectura. Como mínimo, el lector debe estar familiarizado con los fundamentos del sistema operativo UNIX, y sentirse cómodo ejecutando comandos y accediendo a archivos del sistema. Los usuarios que no están familiarizados con estos temas deberían considerar el examen de los otros títulos de esta serie.

Versiones UNIX

Todas las herramientas analizadas y los ejemplos que trae este libro provienen del sistema operativo UNIX Linux. Como muchas de las herramientas UNIX están disponibles en una amplia variedad de versiones de UNIX, el lector tendrá poca dificultad para adaptar y usar las herramientas en su entorno. La mayor parte del software está disponible en forma de paquetes RPM, y se puede encontrar en el CD-ROM que viene con este libro. *Véase* el apéndice B, "Procedimientos de instalación", para obtener un listado completo del software disponible en el CD-ROM, así como la información e instrucciones detalladas para su instalación.

Vista general de este libro

Los tres primeros capítulos proporcionan una visión general al lector y se enfocan en la administración general de la red, el protocolo simple de administración de red y la red TCP/IP. Debido a que este libro ofrece una mirada general a las herramientas de administración de red, sería apropiado incluir información introductoria sobre el componente central de la administración de red.

El capítulo 1, "Vista general de la administración de redes", proporciona una introducción a la administración de red; describe los elementos y las funciones básicos que se encuentran en los sistemas de administración de red y define los componentes específicos a que se hará referencia en el libro. Establece el marco de trabajo básico para la revisión de información adicional y temas avanzados que constituyen los fundamentos básicos. El capítulo 2, "Protocolo simple de administración de red", incluye una detallada descripción del SNMP, que es el protocolo de administración más popular y ampliamente implementado hoy en día. Así

mismo, analiza la operación central de los protocolos y explica cómo tanto los administradores de redes como los agentes operan de una forma perfecta y fluida. El capítulo 3, "Conjunto de protocolos TCP/IP", proporciona una vista general detallada de TCP/IP. Cubre muchos aspectos del trabajo en red, como las direcciones IP y la operación básica de TCP y otros protocolos que un administrador de red o un administrador de sistemas puede necesitar cuando esté usando o administrando sistemas UNIX.

Los dos capítulos siguientes se enfocan en el uso de herramientas de software de UNIX, para la administración individual y de sistemas en red. El capítulo 4, "Herramientas y utilidades centrales del sistema", proporciona una vista general detallada de algunas de las herramientas y utilidades más comunes de UNIX. Incluye ARP, ifconfig, netstat, ping y traceroute. El capítulo 5, "Utilidades y herramientas adicionales del sistema", amplía el tema de los capítulos anteriores, al introducir herramientas y utilidades adicionales, como nmap y ethereal.

El capítulo 6, "Vista general de MIB-II", describe los objetos disponibles en el almacén de objetos MIB-II. Éstos son los objetos implementados en la mayoría de productos de los distribuidores. El capítulo 7, "Uso de los agentes SNMP", introduce los agentes SNMP y sus objetos MIB asociados. Este capítulo ayudará a los lectores a entender los servicios básicos y las funciones de los agentes. El capítulo 8, "Herramientas SNMP", aborda las herramientas específicas que recuperan información de administración de redes de los agentes SNMP. Estas herramientas forman parte de los fundamentos globales para controlar y configurar sistemas de red.

El capítulo 9, "Herramientas habilitadas para la Web", suministra el repaso necesario de herramientas basadas en la Web que ofrecen ampliaciones adicionales en los aspectos de reportes de administración de redes. El capítulo 10, "Panel de control Linux", describe el paquete estándar de panel de control que viene con Red Hat Linux. El capítulo 11, "Herramienta tkined-editor de la red", proporciona una descripción detallada del editor de red Scotty. Esta herramienta de software puede crear mapas de topología de red para el seguimiento y la administración de una variada colección de dispositivos de red.

El apéndice A, "Un vistazo a las herramientas", suministra una visión general de todas las herramientas analizadas en este libro; el apéndice B, "Procedimientos de instalación y notas sobre el software", proporciona procedimientos de instalación

para muchas de las herramientas analizadas en este libro; y el apéndice C, "Glosario para el trabajo en red/expresiones o términos sobre la administración de red", define algunos de los términos más comúnmente utilizados en redes y en la administración de redes.

El libro contiene dos CD-ROM. Uno de ellos contiene la distribución completa de Red Hat 6.1. El otro contiene muchas de las herramientas analizadas en el libro. También contiene la distribución Red Hat Linux 6.1 Powertools.

Convenciones

En este libro se usan las siguientes convenciones:

- Línea de código, funciones, nombres de variables, y cualquier texto que usted vea en la pantalla aparece en un tipo especial de letra a espacio sencillo o `monoespaciado`.
- Nombres de archivo, botón, icono y de menús también se presentan en tipo de letra `monoespaciado`.
- Nombres de programas, utilidades y herramientas aparecen en una fuente normal, ya sea en letras mayúsculas o minúsculas, como exige la industria (por ejemplo, MRTG y ntop).
- Los términos nuevos están en *itálicas*.
- Si se le ha pedido, dentro del cuerpo de un párrafo, que escriba cualquier texto, el texto que debe digitar aparecerá en **negritas**. Por ejemplo, "Escriba **/mysqlshow**". Generalmente, sin embargo, la línea, por sí misma, aparece en negrita, con un tipo de letra monoespaciada, como se muestra en el siguiente ejemplo:

  ```
  ./mysqlshow
  ```

- Los comandos y las palabras clave están establecidos en **negrita**.

Reconocimientos

Muchas gracias a Ginny, mi esposa, por su invaluable asistencia y constantes e ingeniosos gracejos durante el desarrollo de este texto. También a los tres mejores niños a quienes amo más que a nadie y con quienes he madurado: Lisa, Matthew y Joshua. Ustedes han hecho que mi vida valga la pena. El autor desea agradecer al

personal de producción de McGraw-Hill por sus notables esfuerzos al preparar el manuscrito. Agradecimientos especiales para Simon Yates, editor senior, por administrar y creer en la serie de Herramientas UNIX, y por coordinar el bello arte de la publicación de libros. Muchas gracias también a Bob Kern por su gran destreza en la edición de este libro. Mis agradecimientos también para Sandra Henry-Stocker, por leer cuidadosamente el manuscrito durante el proceso de revisión, y a Lynanne Fowle, por la digitación del libro y por manejar todos los pequeños detalles.

Gracias especialmente a JC, sin quien no me hubiera sido posible producir nada, y por hacer todo esto una realidad.

Acerca del autor

En los últimos 15 años, Steve Maxwell ha estado involucrado en el diseño, la instalación y la administración de redes empresariales, en una gran variedad de compañías e instituciones. Él ha ayudado a construir y dar soporte a redes y sistemas para Lisp Machine, BBN, la Universidad de Boston, Stratus Computer, 3Com, y otras. Además, mientras estuvo en 3Com, ayudó a crear productos para la administración de redes y profundizó en los fundamentos y las estrategias necesarios para abordar efectivamente los requerimientos de la administración de grandes redes heterogéneas. Steve Maxwell vive actualmente en Orlando, en el área de Florida, con su esposa y sus tres hijos, creciendo.

Steve recibe con agrado su retroalimentación. Por favor, envíe sus comentarios y sugerencias acerca de este libro a: sjmaxwell@worldnet.att.net

01

Vista general de la administración de redes

En un nivel conceptual, un sistema de administración de red abarca tres áreas fundamentales: seguimiento del rendimiento, administración de la configuración y administración de diagnósticos. Algunos sistemas de administración de redes proporcionan todos estos servicios, mientras que la gran mayoría de las herramientas abordan sólo uno o dos.

Cada una de estas áreas hace parte de un rompecabezas total, como lo muestra la figura 1.1. Un sistema de administración de red debe funcionar en las tres áreas y suministrar las herramientas necesarias para implementar una administración efectiva de la red. Si una de las áreas no es cubierta, el cuadro total de la administración de la red estará incompleto y la solución será inadecuada. Es decir, sin las herramientas que den soporte a las funciones críticas de la administración de la red, todo esfuerzo administrativo de la red será inadecuado. Uno de los objetivos de este libro es abarcar cada una de las áreas de la administración, describiendo las herramientas que implementan una o más de las funciones de la administración de redes. Sin embargo, debido a que algunas de las herramientas que se citan son de dominio público, no todas las áreas serán cubiertas por una herramienta apropiada para la administración de redes.

Figura 1.1 Tres elementos de la administración de red.

La *administración del rendimiento* incluye la capacidad para mantener la vigilancia sobre la eficiencia general o la salud de la red desde la perspectiva de una empresa o de un dispositivo individual. Involucra la recolección y el análisis subsiguiente de los patrones de tráfico para aliviar los cuellos de botella de la red y solucionar los problemas relacionados, como cuando la transferencia de un archivo entre dos sistemas toma más tiempo del que

debería, o cuando vence el tiempo de actualización de una base de datos de un sitio remoto en un periodo determinado. Para resolver estos y otros problemas, se requiere obtener información histórica adecuada sobre el rendimiento y llevar a cabo un análisis de tendencias de conducta en los datos recolectados. Al realizar revisiones del rendimiento de manera proactiva, el administrador de la red o el diseñador pueden planear el crecimiento de ésta de una forma más consistente, estratégica y oportuna.

La *administración de la configuración* incluye el aseguramiento de que cada uno de los dispositivos del sistema contenga la configuración apropiada y la versión correcta del software del sistema operativo. Por ejemplo, supongamos que tenemos que habilitar parámetros de seguridad en un conjunto de dispositivos de red. Algunas veces, los cambios se hacen en dispositivos individuales, mientras que otras los cambios se propagan en una gran cantidad de sistemas. En estos casos, el sistema de administración de la red debe asegurarse de que estos cambios sean hechos de manera correcta y eficiente.

La *administración de diagnósticos* involucra la solución de fallas de la red, problemas de software y otros factores y problemas que afectan la operación normal de la red o de sus componentes. Esto también involucra el manejo efectivo de muchos y diferentes tipos de problemas que resultan de fallas relacionadas con software y hardware. Los problemas de una red varían en complejidad y pueden ir desde una simple falla de hardware hasta un problema relacionado con protocolos. El objetivo final del sistema de administración de una red es aislar, describir y, en algunos casos, solucionar el problema.

Un reto importante en la administración de una red es mantener el ritmo con las mejoras y los avances en la tecnología de redes como un todo. Con la instalación de redes virtuales de área local (VLAN, *Virtual Local Area Networks*), donde las topologías de la red pueden definirse y modificarse muy rápidamente y los dispositivos individuales y las estaciones de trabajo se mueven alrededor de la red en forma más lógica que física, la necesidad de una administración funcional y robusta es crítica. Con la migración de topologías estándar de redes de área amplia (WAN, *Wide Area Networks*), la creación de redes virtuales privadas (VPN, *Virtual Private Networks*), donde una red pública se usa como base de la infraestructura de red, y la eventual implementación a gran escala de redes de área de almacenamiento (SAN, *Storage Area Networks*), la administración de redes

debe desempeñar otra vez un importante papel en la formación de la administración, el soporte y las operaciones diarias de la red.

Con estos factores en mente, presentamos una vista general de los componentes principales del sistema de administración de una red, con énfasis en la funcionalidad específica del sistema.

Componentes de la administración de redes

Para entender cómo funciona un sistema de administración de red, debemos considerar primero los elementos individuales que forman tal sistema. En general, un sistema de administración de red consta de los cuatro componentes siguientes:

- administradores
- agentes
- bases de información de administración (MIB, *Management Information Bases*)
- agentes *proxy*

Los primeros tres componentes son elementos que se requieren en la mayoría de los sistemas de administración de redes. El elemento restante, servicios *proxy* o *proxis*, es opcional porque estos servicios tienden a ser muy especializados y podrían no ser requeridos en la mayoría de ambientes de red.

Administradores

El software de administración de red incluye una importante funcionalidad para ayudar al administrador de la red en las tareas diarias de manejo de la red o de los dispositivos individuales. El software del administrador de red consulta una entidad agente, de manera regular, para recolectar información vital. Un administrador debe *sondear* muchos agentes de manera regular, para recolectar una variedad de información, respecto del estado de operación, la configuración actual o los datos de rendimiento. La figura 1.2 muestra la relación entre un administrador y un agente desde un punto de vista de alto nivel. El administrador usa la información recolectada para determinar la salud general de los dispositivos individuales de la red, una porción de la red, o de la red como un todo.

Figura 1.2 Relación administrador/agente.

Agentes

Los agentes de administración de red son módulos de software que residen en dispositivos de red, como las estaciones de trabajo UNIX, computadoras personales, impresoras de red, enrutadores, conmutadores (*switches*) y otros dispositivos de red. Los agentes tienen acceso al estado operacional, características de dispositivos, configuración del sistema y otra información relacionada, como se muestra en la figura 1.3. Los agentes son corredores (*brokers*) de información para cada dispositivo de la red administrado; ellos responden a solicitudes sobre información específica y son generalmente sondeados por uno o más administradores de red. Los agentes son responsables por la interfaz entre el software de administración y el dispositivo en el cual reside el agente. Los

Figura 1.3 Agente dentro de un dispositivo de red.

agentes pueden controlar las funciones de los dispositivos de red manipulando la base de datos de información almacenada en la base de información de administración (MIB) contenida en el dispositivo.

El comportamiento externo de los agentes de administración de la red es el mismo en los diferentes productos de los vendedores, si el agente soporta un protocolo estándar como el protocolo simple de administración de red (SNMP). Los agentes SNMP entienden tanto el lenguaje como el tipo de preguntas que un administrador puede hacer. Es posible emplear un sistema de administración de red adquirido a un distribuidor, con aplicaciones de red de terceros adquiridas con otro, y agentes desde un *host* de vendedores adicionales, y asegurar la comunicación entre todos los componentes. SNMP provee el marco de trabajo que permite la comunicación y la comprensión entre muchos dispositivos, software y sistemas dispares.

Las partes internas de muchos dispositivos de red, como enrutadores, conmutadores y concentradores (*hubs*), son de propiedad del diseñador, y el componente del agente es suministrado por el fabricante de equipos de red como parte del sistema básico, o como un módulo opcional de actualización. En muchos casos, el agente está incrustado en el sistema operativo del dispositivo y generalmente ofrece opciones mínimas de configuración. Esto está en contraposición con los sistemas operativos de las computadoras, como UNIX, donde el agente es suministrado como un programa de sistema independiente y el usuario tiene mucho más control sobre la operación y la configuración del sistema. Bajo el sistema UNIX, varios agentes están disponibles, tanto de administradores comerciales como de dominio público. En contraste, ningún agente de dominio público está disponible para la mayoría de productos de hardware de red.

Es responsabilidad del agente de administración de la red convertir las solicitudes del administrador de la red desde un formato estándar de red, recuperar la información deseada y enviar de regreso una respuesta válida. En algunas situaciones, el administrador puede solicitar al agente que realice alguna acción asignando el valor del objeto de la base de información de administración (MIB). Por ejemplo, si un agente recibe una solicitud para reiniciar un dispositivo, no se enviará ningún mensaje de retorno, pero el agente tratará de satisfacer la solicitud, siempre que se cumplan los requerimientos de seguridad. Sólo con sondeos adicionales al dispositivo, el administrador sabrá realmente si éste fue reiniciado.

Los vendedores de equipos definen los objetos que sus agentes manipulan o soportan, lo cual plantea algunos problemas interesantes para los desarrolladores de agentes. ¿Cómo determina un vendedor la funcionalidad del agente? Generalmente, el vendedor busca consejo del cliente, y en un mundo ideal, intenta imitar funciones de hardware. Es así como ellos tratan de habilitar al agente para dar soporte a las funciones o los servicios de hardware tan cerca como sea posible, dadas las restricciones de marketing e ingeniería. Sin embargo, esta meta no siempre es práctica o fácil de alcanzar; los diseños de hardware generalmente cambian muy rápidamente, y los software de agentes podrían no seguirles el paso a las mejoras del hardware. Además, la administración de red como un todo está aún en sus inicios, y no todos los vendedores de equipos operan dándose cuenta de que el desarrollo del agente es tan crítico como el desarrollo de productos de hardware. En otras palabras, tener el dispositivo más rápido en el mercado es, en muchas situaciones, más importante que suministrar un agente completamente funcional.

Usualmente, un solo agente opera en un dispositivo de red. Sin embargo, algunos de los más costosos equipos de alta gama pueden tener más de un agente disponible. Existen varias razones para utilizar múltiples agentes. Primero, los múltiples agentes suministran un nivel de redundancia agregada y, por tanto, mayor resistencia contra las fallas. Si uno de los módulos/agentes falla, otro tomará su lugar en la tarea requerida; de este modo, se asegurará la disponibilidad y operación constante del agente. Segundo, agentes adicionales pueden tener un impacto positivo en el rendimiento concerniente al sondeo. Podría ser más eficiente sondear dos agentes independientes, y no sólo uno. Tercero, el diseño o la arquitectura de hardware podría garantizar múltiples agentes independientes.

Sin importar la razón, tener más de un agente introduce problemas y temas únicos para un sistema de administración de red. Por un lado, los múltiples agentes pueden aparecer para el administrador como dispositivos diferentes. Esto puede causar problemas para la aplicación que recolecta información de inventarios, porque el número de dispositivos registrados por la aplicación difiere del número real de dispositivos de hardware instalados en la red. Los múltiples agentes también obligan al administrador de la red a ejecutar trabajos con todos los agentes en mente, no sólo uno, aumentando así la complejidad de la administración de la red.

Los agentes tienden a ser muy estáticos por naturaleza; ellos soportan un conjunto fijo de funciones u objetos. En muchos

dispositivos de red, los agentes pueden ser modificados o actualizados sólo por el vendedor del dispositivo. Esto es, de alguna forma, desafortunado, porque es de mucha ayuda añadir funcionalidad adicional al agente, sin esperar a que el vendedor suministre las actualizaciones. Sin embargo, un pequeño número de agentes son extensibles y se pueden configurar para soportar funciones adicionales. En general, los agentes que residen en el sistema UNIX o en otros sistemas operativos tienen uno o más archivos que controlan algunos de los aspectos operacionales o de configuración del agente. El agente del sistema que está disponible para Linux, por ejemplo, es muy configurable. Muchos agentes extensibles están disponibles sólo para sistemas operativos genéricos, como UNIX, frente a dispositivos específicos como los enrutadores Cisco o conmutadores de canal de Brocade. Los agentes extensibles proporcionan al usuario la capacidad de conectarle a nuevas posibilidades, sin recurrir a la construcción o actualización de otra versión del agente. La figura 1.4 muestra una vista funcional de este tipo de agente. Como puede verse, el componente extensible es simplemente una adición para el agente. En muchos casos, requiere parámetros y comandos adicionales de configuración para ser agregados al archivo de configuración del agente.

Figura 1.4 Diagrama funcional de agente.

Base de información de administración (MIB)

La MIB define una base de datos de objetos que pueden ser recuperados o manipulados por un sistema de administración de red. La figura 1.5 muestra la relación del agente y los objetos MIB. La MIB es una bodega de información y puede contener literalmente miles de objetos que el administrador de la red puede usar para

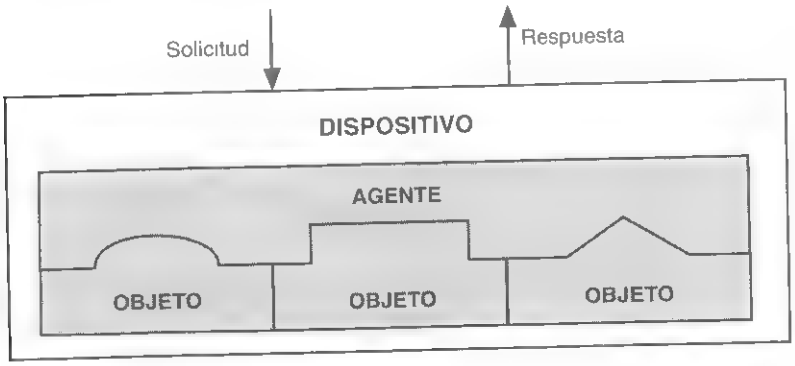


Figura 1.5 Agente SNMP y objetos MIB.

controlar, configurar y hacer seguimiento a los dispositivos, manipulando directamente los objetos. Los objetos MIB están a disposición del sistema de administración de redes, a través del agente. Es responsabilidad del agente mantener la consistencia de los objetos MIB, sin tener en cuenta el número de los objetos.

Varios de los estándares comunes MIB se han definido, lo cual incluye objetos específicos que deben estar soportados en dispositivos de red, de tal forma que sean compatibles con el protocolo SNMP. La MIB más ampliamente implementada y común es la MIB-II. Muchas MIB adicionales se han desarrollado y documentado en las Solicitudes de Comentarios (RFC) para dirigir diferentes componentes y tecnologías de trabajo en red. La tabla 1.1 muestra una lista de ejemplos de estas MIB. Como se puede ver, las MIB están disponibles para una diversa colección de servicios, protocolos de red y arquitecturas.

Adicionalmente, muchos vendedores han desarrollado sus MIB propietarias. Estas MIB definen objetos adicionales que no se encuentran en las MIB estándar. Estas MIB adicionales son necesarias para lograr una ventaja plena de las funciones de administración de un vendedor específico, que puedan estar disponibles. Como las MIB están escritas en un lenguaje estándar, pueden usarse con muchas herramientas y productos diferentes para la administración de redes. Muchos vendedores de equipos de red como 3Com, Brocade, Cabletron y Cisco, así como fabricantes de computadoras como HP, IBM y Sun, han desarrollado sus propias MIB. La tabla 1.2 muestra un ejemplo de MIB propietarias que están disponibles a través de algunos de estos vendedores.

Tabla 1.1 Lista de MIB estándar.

MIB	Descripción
FDDI MIB	Suministra acceso a las funciones de administración de red de la interfaz de distribución de datos por fibra (FDDI)
Recurso *host* MIB	Suministra acceso a los objetos del sistema que están relacionados con la memoria, los discos y los procesos, y a otra información relacionada con el sistema
ISDN MIB	Se utiliza para administrar los nodos de la red digital de servicios integrados (ISDN)
Solucionador DNS y servidor MIB	Provee capacidades de administración para el servicio de nombres de dominio (DNS)
MIB-II	Suministra acceso a objetos en el sistema, que incluyen protocolos de red, interfaces de red e información genérica del sistema
MIB de impresora (*printer MIB*)	Suministra acceso a las funciones comunes de administración de la impresora

Tabla 1.2 Lista de MIB de propiedad exclusiva o propietaria.

MIB	Descripción
3cProd-mib	MIB de sistemas 3COM para la fabricación de productos centrales de conmutación
3cWeb-mib	MIB servidor de Web de 3Com
Cisco-mib	MIB enrutador de Cisco
snmpdx.mib	MIB agente maestro de Sun
swmib	MIB de conmutador Brocade

Agente *proxy*

Un agente *proxy*[1] acorta la distancia entre los administradores de protocolos estándar de red y los sistemas heredados que no soportan directamente tales normas. El término agente *proxy* también se puede referir a un agente que soporta más de un protocolo de administración de red. La figura 1.6 muestra las funciones básicas de un sistema basado en un *proxy*. Los agentes *proxy* proporcionan un camino de migración para protocolos estándar antiguos a nuevas versiones, sin necesidad de actualizar toda la red. Por ejemplo, en los últimos años, una significativa cantidad de trabajo se ha enfocado hacia el establecimiento de mejoras del protocolo SNMP.

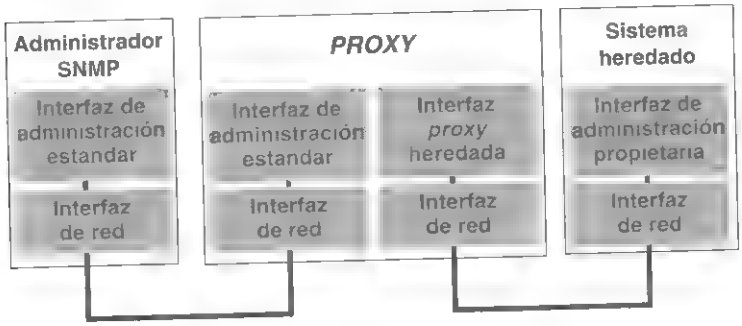


Figura 1.6 Sistema basado en *proxy*.

Cuando las nuevas versiones de SNMP estén disponibles, será deseable que los dispositivos de red existentes estén actualizados. Sin embargo, cuando estas mejoras estén disponibles por parte de los vendedores de equipos, es razonable suponer que no todos los productos estarán actualizados para soportar los nuevos protocolos. Algunos vendedores distribuirán actualizaciones de sus productos, y otros no; es un problema de viabilidad del producto y de economía. Es posible que se actualicen dispositivos críticos, como los enrutadores de alta calidad, mientras que otros productos se vuelvan obsoletos. Como resultado, las organizaciones deben abordar el problema de soportar diferentes versiones SNMP para un futuro previsible. El mecanismo *proxy* es una de las formas de

1. El término *proxy* como se emplea aquí no hace referencia a los servidores *proxy* usados actualmente en muchas redes empresariales que suministran filtrado de protocolo, seguridad y otros servicios relacionados para ocultar los detalles de una red interna al mundo exterior.

tratar con las preocupaciones de migración, porque puede ser usado como un puente con dispositivos antiguos no actualizables.

Es interesante notar que un sistema basado en *proxy* puede también proporcionar conectividad de red a sistemas que no soportan ningún protocolo de red estándar (como TCP/IP) y, por tanto, suministra también servicios de conectividad básicos de red.

Funciones del administrador

Hoy en día los administradores ofrecen un conjunto de servicios para abordar las necesidades de una red empresarial heterogénea en expansión. Los servicios que proporciona pueden venir del mismo sistema de administración básico o de una aplicación de terceros, o pueden ser desarrollados dentro de la misma empresa para abordar algunos requerimientos únicos específicos. Las habilidades y funciones del administrador de la red pueden ser agrupadas en tres áreas:

- arquitectura
- servicios centrales
- aplicaciones

La tabla 1.3 enumera los servicios asociados para cada una de las categorías, y a continuación sigue un análisis sobre los servicios de las categorías.

Tabla 1.3 Servicios del administrador enumerados por categorías.

Categoría	Arquitectura	Servicios centrales	Aplicaciones adicionales
Marco de trabajo abierto/extensible	✔		
Seguimiento centralizado/ distribuido	✔		
Plataformas comunes	✔		
Detección de fallas/solución		✔	

Tabla 1.3 Servicios del administrador enumerados por categorías. (*continuación*)

Categoría	Arquitectura	Servicios centrales	Aplicaciones adicionales
Notificación de alarma		✔	
Gran número de dispositivos		✔	
Herramientas de reporte		✔	
GUI de uso fácil		✔	
Administración de configuración		✔	
Descubrimiento de red		✔	
Distribución de software			✔
Etiquetado de problemas			✔
Administración de políticas			✔
Procesamiento avanzado de alarma			✔
Simulación de red			✔

Arquitectura

La arquitectura describe el marco de trabajo básico y el modelo de cómo el administrador opera e implementa los servicios básicos necesarios para cumplir con las tareas de la administración de red. Ésta debe incluir:

Un marco de trabajo abierto y extensible: algunos administradores están diseñados alrededor de un marco de trabajo basado en estándares abiertos, el cual puede soportar ampliaciones de

protocolos existentes y tecnologías emergentes. Un administrador de red abierta soporta los protocolos de administración de red basados en los estándares, como SNMP y el protocolo común de información para la administración (CMIP). En la actualidad, el protocolo de administración de red preferido es el SNMP. Un sistema de administración de red que no soporte una arquitectura abierta y los protocolos estándar debe verse con suspicacia porque puede ser difícil para el software soportar ampliaciones en los protocolos de red, bases de datos y software del sistema operativo. Los administradores deben soportar también el conjunto de protocolos de Internet TCP/IP y otros protocolos de trabajo en red propietarios.

Un sistema abierto de administración proporciona también herramientas de integración e interfaces de programación de aplicaciones (API) para que otros desarrollen soluciones adicionales de software al marco de trabajo de administración. Estas soluciones suministran funciones especializadas de aplicación que usualmente no están disponibles en el sistema de administración de red básico. Dependiendo del nivel de integración, el usuario puede no estar consciente de las diferentes soluciones de los vendedores para el sistema de administración.

Soporte para el seguimiento distribuido/centralizado: los sistemas de administración de red deben tener capacidad de instalarse en una arquitectura distribuida o centralizada. Esto se conoce también como el enfoque cliente/servidor, y suministra los medios para que el sistema de administración de red aumente y direcciones los requerimientos adicionales de la administración de la red. En muchas situaciones es recomendable instalar uno o más sistemas de computadoras dedicadas a fin de manejar la sobrecarga relacionada con la administración de una red de gran tamaño. Aunque los vendedores no siempre requieren de más de un sistema para las tareas de administración, es irracional suponer que un solo sistema de administración de red puede manejar y soportar una red que conste de decenas de miles de dispositivos. Si, hipotéticamente, la plataforma seleccionada pudiera detectar esa cantidad de dispositivos, algunas consideraciones prácticas justificarían estar en contra del uso de un solo sistema.

Primero, en una aproximación (enfoque) centralizada, el administrador debe sondear cada dispositivo desde un solo segmento o subred. Esto puede ocasionar serias congestiones de tráfico para esa porción de la red y reducir significativamente la capacidad de los otros dispositivos en ese segmento para comunicarse con el resto de la red. Segundo, el sistema de administración de red debe estar tan ocupado con la tarea de sondeo que no será

capaz de responder a la GUI o a otras tareas de administración relacionadas. Tercero, el uso de un sistema centralizado es un problema respecto a la disponibilidad general y al tiempo disponible. Si la principal estación administrativa falla debido a un problema de hardware o software que entrega el sistema inservible, la red estará sin una plataforma de administración hasta que el sistema sea reparado. Esto podría ser una situación inaceptable en ambientes de computación donde la disponibilidad del sistema es crítica.

Soporte para plataformas comunes: para proporcionar más opciones a los clientes, los vendedores de sistemas de administración de red suministran un soporte de plataforma cruzada en los sistemas operativos y plataformas de hardware más populares. Muchos vendedores soportan tanto los sistemas operativos UNIX como Windows NT. Sin embargo, UNIX es aún la plataforma más popular para los vendedores de sistemas de administración de red y continuará desempeñando un papel muy importante en el futuro. Las plataformas UNIX más populares entre los vendedores de software de administración de red incluyen a Solaris, AIX y HP-UX. Debido al tremendo crecimiento de Linux, programas de administración de red, como el producto Open View de Hewlett-Packard (HP), han sido llevados a la versión Red Hat de Linux. Es imposible predecir de manera precisa en este momento el papel que desempeñará Linux en el mercado de administración de red. Una cosa es segura: Linux es una solución robusta, poderosa y efectiva en cuanto a su costo, que será implantada en muchas áreas de la computación y del trabajo en red en un futuro previsible.

Servicios centrales

Un sistema de administración de red suministra usualmente un nivel fundamental de servicios. Estos servicios básicos son considerados típicamente como la cantidad mínima de funcionalidad que un sistema de administración de red necesita ofrecer para cumplir tareas simples de administración. Por tanto, debe suponerse que estas capacidades están disponibles en la mayoría de los sistemas empresariales de administración. En algún grado, los vendedores compiten en la industria suministrando ofertas más significativas de servicios centrales y asegurando la disponibilidad de productos de terceros. Los servicios pueden ser suministrados con el sistema básico o como componentes opcionales adicionales. Los servicios básicos adicionales pueden incluir:

Detección y resolución de fallas: las fallas de red son problemas o condiciones que ocurren en la red. Esto incluye problemas

de configuración, dispositivos con sobrecarga, dispositivos que han caído debido a problemas de software o hardware, y muchos otros problemas. Debido al gran tamaño y complejidad de muchas redes empresariales, el software de administración debe suministrar un mecanismo para detectar, reportar, filtrar y, en algunas situaciones, solucionar fallas y problemas en la red. Por ejemplo, cuando un sistema crítico agota el espacio disponible del disco, el sistema de administración de red debe intentar abordar el problema ejecutando una utilidad del sistema para eliminar los archivos innecesarios. Además, como en la mayoría de redes empresariales, el número de fallas de red es directamente proporcional al número de dispositivos monitoreados, el cual puede ser bastante grande. El manejo inadecuado de estas fallas puede impactar la habilidad del personal de soporte de red para responder al problema.

Notificación y procesamiento de alarmas: junto con la detección y resolución de fallas, el administrador debe suministrar un mecanismo para responder a errores y condiciones detectadas. Una alarma es una respuesta programada en relación con una condición de la red. Las alarmas conectan la condición de la red a alguna acción que abordará el problema. Por ejemplo, una alarma es generada cuando un servidor crítico deja de estar disponible para la red, y activa el localizador (*pager*) del administrador del sistema, suministrando la notificación de la falla. Por otra parte, los eventos son condiciones que no siempre se consideran críticas y pueden no garantizar una notificación o reacción inmediata. Un cambio en la configuración de un dispositivo o una disminución en la carga de un servidor se puede considerar un evento de red. Para eventos como éstos, debería ser razonable simplemente registrarlos para una revisión futura.

Soporte de gran número de dispositivos: los administradores están diseñados y son solicitados para manejar un número significativamente grande de dispositivos activos. Con frecuencia, un administrador da soporte a cientos o incluso miles de dispositivos, desde un punto de vista teórico. No obstante, en la práctica, los administradores no están configurados para sondear cantidades muy grandes de dispositivos, debido a consideraciones de rendimiento de la red.

Herramientas de reporte: uno de los requerimientos más significativos de un sistema de administración de red es su capacidad para producir reportes efectivamente. Los tipos de reporte considerados esenciales incluyen rendimiento, configuración, inventario y falla. En muchos casos, el área más crítica para reportar es

la relacionada con el rendimiento. Los sistemas de administración de red pueden recolectar una gran cantidad de estadísticas de rendimiento. Se requiere una facilidad flexible y de uso sencillo para abordar con determinación el bienestar general de la red; así, esos cuellos de botella potenciales pueden ser señalados rápida y fácilmente. Además, el software de reporte debe manejar requerimientos de inventario, con el fin de asistir, por ejemplo, con información de revisión de un dispositivo determinado, necesario para ayudar con las actualizaciones de la red.

Interfaz de fácil uso: típicamente, un administrador tiene una interfaz de usuario gráfica (GUI) que facilita las tareas y funciones de la administración. En la actualidad, los sistemas de administración de red están experimentando una transformación respecto a las interfaces. Muchos vendedores de sistemas de administración de red suministran tanto el soporte nativo de GUI como interfaces basadas en la Web, para sus productos. Algunos vendedores han tomado el concepto GUI en su máximo potencial, y muchos han sido exitosos uniendo dispositivos dispares de red para compartir una vista uniforme de la administración y el control. Esto ha sido una gran ventaja en la administración de redes desde un punto remoto o *help-desk*. El modelo de la red ciertamente cambiará la forma tradicional en que se ha realizado la administración de redes. Sin embargo, podrá transcurrir algún tiempo antes de que los productos de administración de red sean migrados completamente a una interfaz habilitada para la Web.

Seguimiento y medición del rendimiento: una de las áreas más importantes que causan problemas en los administradores de red es el manejo del rendimiento de la red. Algunos vendedores de sistemas de administración de red suministran herramientas para recolectar y analizar la información del rendimiento de la red. Esta información es vital en el diagnóstico de los problemas relacionados con el rendimiento y ayuda en la planeación proactiva de las actualizaciones de la red o/y otras ampliaciones para mejorar las características de rendimiento de la red. Las herramientas de rendimiento son asistidas por medio de algunos productos de hardware especial (y algunas veces software) que recolectan información de rendimiento de manera continua. Estos dispositivos, que se conocen comúnmente como sondas de red, son instalados o incrustados con otros dispositivos como los enrutadores y los conmutadores. Estas sondas escuchan en la red y recolectan información de rendimiento y comunicación de paquete entre los nodos de la red. Las herramientas de rendimiento

de red interactúan con estos dispositivos, inundan las sondas para datos de rendimiento y suministran reportes y capacidades gráficas robustos.

Administración de la configuración: las redes empresariales constan a menudo de equipos que requieren información de configuración personalizada para controlar la funcionalidad y corregir las operaciones del sistema. Un sistema de administración de red debe suministrar soporte para cambiar la configuración de los dispositivos. Por ejemplo, con frecuencia se actualizan uno o más parámetros de configuración a un número mayor de dispositivos. En este caso, sería ventajoso para el administrador manejar esta tarea de una manera lógica y uniforme. Los sistemas de administración de red son criticados a menudo por la falta de soporte de administración de configuración sofisticado. Sin embargo, se están logrando enormes progresos para abordar la compleja tarea de la administración de configuración y su soporte.

Descubrimiento de la red: debido a la magnitud del volumen de dispositivos contenidos en una red empresarial, es irracional agregar manualmente todos los dispositivos a un sistema de administración de red. Por tanto, los administradores suministran la capacidad para buscar o *descubrir* de manera automática una red entera o una clase especial de dispositivos. El proceso de descubrimiento de la red generalmente toma tiempo y consume una cantidad significativa de ancho de banda de la red. Este proceso se debe realizar con cuidado, ya que puede afectar adversamente el rendimiento de la red. Después de finalizar el proceso de descubrimiento, los dispositivos son puestos en vistas especiales, las cuales son con frecuencia arregladas por subredes.

Aplicaciones o servicios adicionales

A menudo, los sistemas de administración de red son desplegados en donde los servicios y las funciones adicionales son necesarios para direccionar los procesos del negocio y soportar la estructura de la organización. Estos servicios extenderán la utilidad y capacidad de un administrador de red básico suministrando aplicaciones de valor agregado que son puestas sobre el sistema de administración de red. Estas aplicaciones pueden estar disponibles a través de terceros o del mismo vendedor del sistema de administración de red. En muchos casos, los vendedores de administradores de red buscarán socios como solución, para suministrar software específico de negocios, que podrá complementar e integrar el sistema de administración de la red.

El nivel de integración de productos de terceros dependerá de los servicios centrales del sistema de administración de red, del esfuerzo involucrado en la integración del sistema de administración y en la funcionalidad del producto. Dependiendo del sistema de administración, la integración a partir de un vendedor puede significar el tener la capacidad de recurrir a funciones de software desde un menú común, o puede significar ganar acceso a la base de datos.

Algunas de las aplicaciones y servicios más populares que están disponibles incluyen:

Distribución de software: en ambientes grandes de computación es necesario tener un mecanismo automatizado para mantener el software a través de muchos sistemas de computación que se encuentran en la red. En muchos lugares, la red contiene computadoras personales, estaciones de trabajo y servidores, cada uno soportando uno o más sistemas operativos diferentes. Para abordar el proceso de administración del software para esos sistemas, los sistemas de administración de red suministran software automatizado de distribución de software que puede ser utilizado para cargar parches del sistema operativo y software de aplicaciones con sólo oprimir un botón. Imagine, por ejemplo, tener que instalar una aplicación de terceros en cien sistemas. Sin un software de administración de red para *empujar* la aplicación a estos sistemas, la única alternativa es instalar este software de forma manual o construir una secuencia de comandos (*script*) de instalación personalizada para hacer el trabajo. Llevar a cabo esta tarea manualmente requiere una cantidad significativa de tiempo y esfuerzo. Debido a que los administradores de red ya tienen mucho que hacer, el software de administración puede contribuir de forma positiva a solucionar esta situación.

Etiquetado de problemas: esta aplicación suministra habilidad para seguir y administrar las fallas y los eventos de la red en forma automática. Cuando el etiquetado de problemas es integrado con un sistema de administración de red, ciertas alarmas y eventos de red pueden desencadenar la creación de una etiqueta que notificará el problema, individual o en grupo, propio de la red. Un problema, con un enfoque automatizado, consiste en la posibilidad de crear etiquetas indeseadas o no válidas. Por ejemplo, una edificación podría quedar sin energía momentáneamente y reiniciar porciones de la red; en este caso, podrían generarse muchas etiquetas porque una gran porción de la red no estuvo disponible. Sería suficiente generar una sola etiqueta, en lugar de una etiqueta para cada uno de los dispositivos. Algunos sistemas de etiquetado de problemas tienen la habilidad de filtrar el ingreso de alarmas

basado en el criterio definido por el usuario, o en otras condiciones. Como resultado, se puede crear un filtro para medir la duración del apagón y, sólo después de establecer el periodo transcurrido, generar la etiqueta apropiada. Cuando se usa con cuidado, este filtro puede suministrar una manera completamente automatizada de enviar personal clave cuando ocurran problemas en la red, sin la necesidad de un centro de soporte de llamadas exclusivo.

Administración basada en las políticas: la administración de políticas es un aspecto relativamente nuevo en la administración de red. Una política involucra el establecimiento del perfil de un recurso o dominio de acceso para un usuario individual, un grupo o un sistema. También puede contener información del usuario, como los derechos de acceso, privilegios, identificación de correo electrónico, y otra adicional. En particular, las políticas pueden utilizarse para definir la red desde listas de control de acceso (ACL) hasta dominios de difusión (*broadcast domains*). Un dominio de difusión describe el área o número de dispositivos con los cuales un sistema se puede comunicar. Las redes virtuales de área local (VLAN) y las redes virtuales privadas (VPN) hacen uso de las políticas para ayudar a definir o formar la red y establecer y hacer cumplir los acuerdos relacionados con el servicio (SLA).

Procesamiento avanzado de alarma: algunas veces, el manejo de la alarma básica del producto central de administración no suministra la personalización o funcionalidad necesarias para abordar los requerimientos específicos. Como resultado, muy probablemente pueden requerirse capacidades más avanzadas de procesamiento de alarmas. Por ejemplo, puede ser necesario que el software de alarma suministre mejores mecanismos de notificación de alarma. Algunas veces no es apropiado activar un localizador, sobre todo en una operación automatizada de apagado de luces. En cambio, el software de alarma debe ser suficientemente inteligente para tomar decisiones acerca de cómo resolver el problema de la mejor manera. Esto puede incluir la ejecución de un programa externo, la secuencia de comandos o la iniciación de una secuencia de terminación de fallas.

Simulación de red: el software de simulación de red emula el comportamiento de un entorno de red real. En ciertas situaciones, es recomendable modelar un diseño de red o actualización antes de realizar algún cambio en la red existente. Mediante la interacción con un sistema de administración de red, las simulaciones de red pueden ser más realistas y dinámicas porque están basadas en información acerca de los dispositivos actuales y de la topología de la red, en lugar de una información estática y genérica.

02

Protocolo simple de administración de red

El protocolo simple de administración de red (SNMP) proporciona el marco de trabajo de bajo nivel para muchos sistemas de administración de red. El SNMP ha sido ampliamente implementado y se puede encontrar en una gran variedad de sistemas y dispositivos de red. En la actualidad, el SNMP está considerado como el protocolo de administración favorito de vendedores de hardware de red, vendedores de administración de red, desarrolladores de aplicaciones de software, proveedores de servicios de Internet (ISP), y usuarios finales.

¿Por qué SNMP es tan popular? Primero, el SNMP es sencillo de implementar, comparado con otras arquitecturas o protocolos de administración de red. El protocolo, las MIB y los marcos de trabajo asociados pueden ejecutarse en cualquier computadora personal común y corriente (PC), hasta supercomputadoras, servidores y dispositivos de red, como enrutadores y conmutadores. Un agente componente de SNMP no necesita ocupar una gran *huella*, en términos de memoria, y no necesita un significativo poder de procesamiento. El SNMP puede generalmente ser desarrollado con mucha rapidez en los sistemas objetivo, aumentando así el tiempo para el mercadeo de nuevos productos y ampliaciones. Cuando SNMP se introdujo al mercado por primera vez, otros mecanismos de administración estaban disponibles, pero el SNMP demostró ser más flexible y fácil de implementar. Es cierto que el SNMP carece de ciertas características que se encuentran en otros protocolos de administración de red (como OSI, por ejemplo), pero su diseño simple, extensibilidad y facilidad de uso minimizan cualquier inconveniente posible.

Segundo, el SNMP es de libre distribución y de dominio público. Por tanto, ningún vendedor puede reclamar la propiedad del protocolo, ni puede ser protegido para evitar que sea copiado por parte de una compañía o un individuo. La única forma de influenciar o cambiar el SNMP es comprometiéndose en el proceso de estandarización de la Fuerza de Trabajo de Ingeniería de Internet (*Internet Engineering Task Force*, IETF), lo cual puede ser una tarea intimidadora. La IETF es una de las agencias de Internet. Los vendedores pueden decidirse a hacer cambios propietarios a SNMP. Tales cambios pueden resultar inútiles, sin embargo, debido a que estos vendedores deben hacer cabildeo frente a otros vendedores y usuarios para que soporten sus ampliaciones no estándar, lo que frustra el propósito de cumplir un estándar, en primer lugar.

Tercero, el SNMP está bien documentado (a través de RFC, artículos y libros de texto) y es bien entendido en la industria. Esto proporciona una base firme para un mejoramiento y una adopción continuos.

Finalmente, el SNMP puede usarse para controlar una variedad de dispositivos. Está incluso encontrando su espacio en equipos no tradicionales, como sistemas telefónicos, equipos de control ambiental, y casi cualquier otro elemento que pueda agregarse a una red y que requiera administración o control.

Operación del SNMP

El SNMP define el formato del paquete y el intercambio de información entre un administrador de red y sus agentes asociados. En su núcleo, el SNMP manipula objetos dentro de la MIB del agente y, como resultado, puede manejar una variedad de tareas definidas en el agente. El protocolo SNMP y sus componentes relacionados están descritos en varios RFC. Cualquier agente que cumpla con el SNMP puede comunicarse con cualquier sistema de administración de red que soporte el SNMP. El sistema de administración es responsable de hacer preguntas sobre los agentes. Esto es conocido también como sondeo de agente. Si el agente soporta MIB estándar, entonces el sistema de administración simplemente solicita uno o más objetos del agente. Si el agente soporta MIB no estándar (por ejemplo, MIB específicas de un vendedor), entonces el administrador debe tener una copia de la MIB del agente para interpretar correctamente los objetos del agente soportados.

Una de las razones por las cuales el SNMP es considerado *simple* es porque suministra tres operaciones de propósito general que pueden ser aplicadas a objetos agente. Estas operaciones o funciones son el núcleo del SNMP; ellas son: *set*, *get* y *trap*.

Set: un sistema administrativo puede actualizar o cambiar el valor de un objeto que está contenido en un agente. La operación *set* es un comando privilegiado porque, por ejemplo, puede usarse para alterar la configuración de un dispositivo o controlar su estado de operación.

Get: un sistema de administración puede obtener o leer el valor de un objeto que está contenido en un agente. La función *get* es la operación más común del SNMP porque éste es el mecanismo primario usado para obtener información administrativa desde los dispositivos de red.

Trap: un agente puede enviar un mensaje no solicitado a un administrador de red. El propósito del servicio *trap* es notificar al sistema de administración de la red de una condición especial o problema, sin que específicamente el sistema administrativo solicite esta información.

El SNMP define la relación y el flujo de mensajes entre administrador y agente, respecto a las comunicaciones, como se muestra en la figura 2.1. Como se puede observar, en la mayoría de

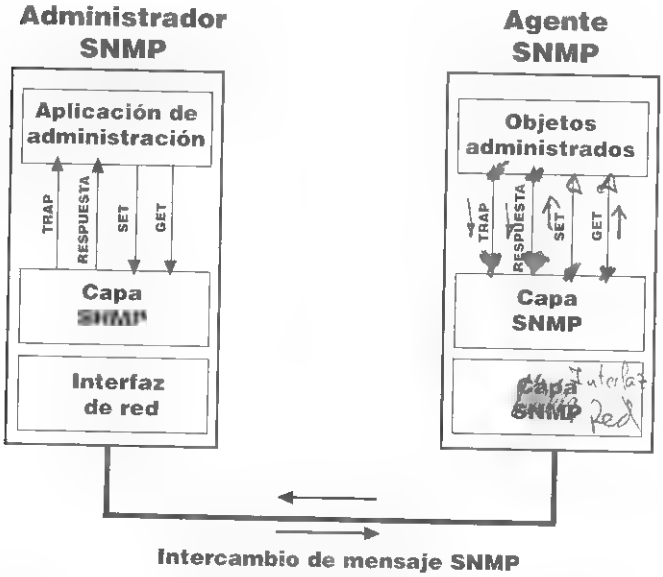


Figura 2.1 Relación administrador y agente SNMP.

los casos, el administrador SNMP dirige el intercambio de mensaje con el agente. Esto se consigue a través de la función *get* o *set*. Una aplicación administrativa hace una solicitud de información; el mensaje se traduce a SNMP usando la capa SNMP, lo pasa a la capa interfaz de red y luego lo transmite a la red. El agente recibe el mensaje en la capa interfaz de red, pasa el mensaje al SNMP para su traducción y tiene acceso a la información deseada del objeto. El proceso contrario se usa cuando el agente envía la respuesta desde el administrador.

El SNMP suministra la capacidad al agente de enviar mensajes asincrónicos al administrador, sin ser específicamente solicitados. Esto se conoce como *trap*. El *trap* suministra a los agentes una forma conveniente e importante de comunicarse sin haber sido interrogados, y también de comunicar información importante o circunstancias o eventos especiales.

SNMP no define ningún mecanismo adicional, fuera de los comandos SNMP enumerados anteriormente, para controlar o emitir comandos para los agentes de dispositivos. Las únicas acciones que se pueden aplicar a los objetos MIB son para establecer (*set*) u obtener (*get*) un valor. Por ejemplo, no hay función de reinicio (*reboot*) definida en SNMP. En cambio, los agentes de software pueden implementar comandos específicos asociando

objetos MIB con los comandos internos soportados por el dispositivo. Para reiniciar un dispositivo, la estación de administración alterará (a través de la operación *set*) un objeto específico MIB a 1, por ejemplo. Éste le indicará al agente que reinicie el dispositivo y restablezca el objeto MIB a su estado previo. Nótese que ningún conjunto estándar de interfaces (u objetos MIB) que esté actualmente disponible puede manipular todos los aspectos de control de dispositivos. En cambio, cada vendedor es responsable de suministrar este acceso usando su propia colección de objetos MIB.

Base de información de administración (MIB)

Como se mencionó anteriormente, la MIB es un almacén de información, relacionada con la configuración, rendimiento y cualquier otro dato de un agente. Las MIB tienen una organización y estructura común y contienen contener un gran número de objetos separados en grupos.

Organización

Los objetos MIB están organizados en una estructura jerárquica en árbol, en la cual cada rama tiene un nombre y un identificador numérico únicos. La figura 2.2 muestra la organización estándar MIB desde la raíz hasta las ramas adicionales. Las ramas del árbol sirven como una agrupación lógica de objetos relacionados. Las hojas o *nodos*, como son llamados a menudo, representan objetos individuales. También se forman subárboles, y representan tanto ramas adicionales intermedias como hojas de conexión. Los objetos dentro de una MIB pueden ser referenciados especificando cada uno de los identificadores numéricos, empezando por la copa del árbol (o raíz) y dirigiéndose hacia la hoja individual u objeto. La raíz de la rama se escribe comúnmente con un punto (*dot*). Acceder a las MIB es similar a referenciar archivos en el sistema operativo UNIX. Sin embargo, una diferencia clave es que los nombres de ruta UNIX pueden expresarse en términos absolutos o relativos. A los objetos MIB se puede tener acceso sólo de una manera absoluta; el formato relativo no está disponible.

Por ejemplo, la figura 2.2 muestra `root(.)` como la posición mas alta del árbol y `sysDescr(1)` como la hoja. El método común para expresar identificadores de objeto es usar la notación punteada (*dotted notation*). Esta notación requiere que un punto separe cada nombre de rama o identificador. A menudo, la raíz (.)

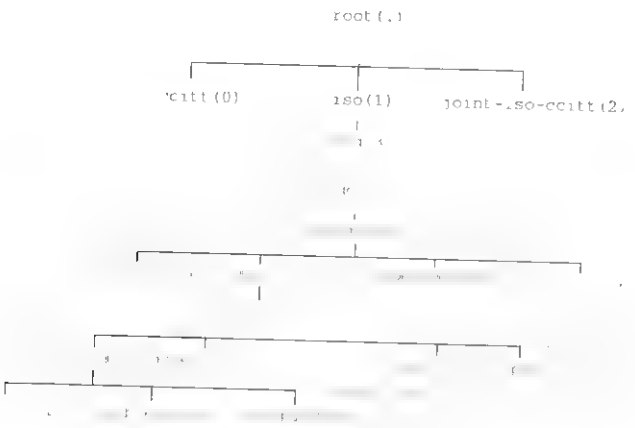


Figura 2.2 Estructura y organización MIB.

no se expresa, está implícita. Para acceder al objeto `sysDescr(1)`, una identificación completamente calificada se escribirá así:

```
iso.org.dod.internet.mgmt.mib-2.system.sysDescr
```

Este identificador se lee de izquierda a derecha. Los objetos pueden también ser expresados de una forma corta, sustituyendo el nombre de la rama por un identificador numérico asociado con cada nombre de identificador. Así, `iso.org.dod.internet.mgmt.mib 2.system.sysDescr` puede también expresarse como `1.3.6.1.2.1.1.1`. Estas dos expresiones son funcionalmente equivalentes y se refieren al mismo objeto MIB. La razón para escoger una forma en lugar de la otra es cuestión de preferencia, aunque el identificador numérico es mucho más conciso. Sin embargo, exploradores MIB pueden mostrar objetos MIB en cualquier formato, haciendo fácil de esta manera convertir de un formato a otro. Después de usar SNMP y MIB por un corto tiempo, usted se familiarizará con ambos métodos y tendrá la oportunidad de escoger el estilo que le convenga.

Al usar la estructura que se muestra en la figura 2.2, un administrador SNMP (o herramienta de exploración MIB) puede ver dentro de la MIB de manera fácil y concisa. Un explorador MIB es una aplicación de software que cruza un árbol un MIB, mostrando generalmente una representación gráfica de ramas, objetos hoja y valores asociados. La figura 2.3 muestra una instantánea de una pantalla de un explorador MIB. Los exploradores MIB son

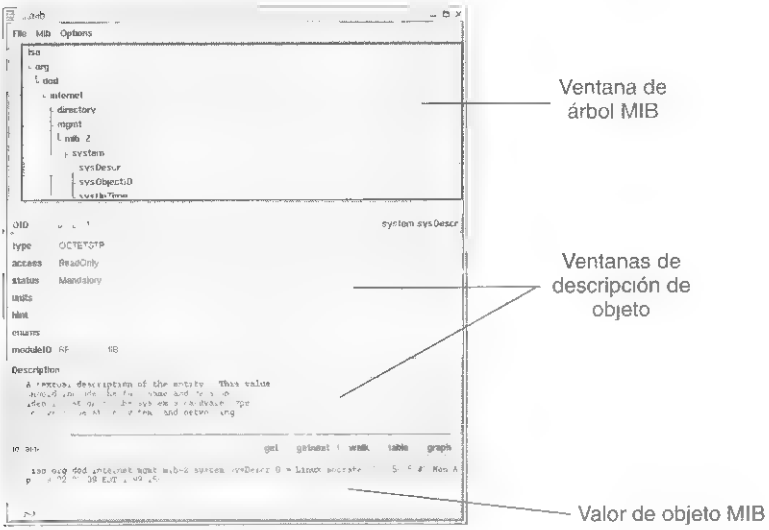


Figura 2.3 Explorador MIB.

estupendos para sondear un agente en busca de información específica, o para aprender la estructura y el formato de nuevos MIB.

La ventana del explorador MIB contiene varias secciones, como se muestra en la figura 2.3. El primer panel suministra una vista jerárquica de las MIB que pueden usarse para *navegar* hacia el objeto MIB o rama deseado. El siguiente panel muestra detalladamente la información respecto al objeto. El tercero muestra el valor obtenido desde el objeto MIB especificado. Información más detallada respecto a esta herramienta puede encontrarse en el capítulo 8, "Herramientas SNMP".

Tipos de objeto

En una MIB, diferentes tipos de objetos representan estructuras de datos o valores contenidos en un agente. Los objetos pueden representar atributos físicos del agente, información de configuración u otros datos. Estos tipos de objetos se derivan de las reglas estándar de la "Notación uno de sintaxis abstracta" (ASN.1). ASN.1 suministra un mecanismo detallado para la implementación y codificación de tipos de datos básicos que son independientes de la máquina y pueden transmitirse por la red de una forma no ambigua y clara.

¿Por qué esto es importante para el SNMP o para la administración de red? Como la administración de red debe abordar un ambiente heterogéneo de red, debe ser provista de una manera estándar para asegurar que los mensajes SNMP puedan ser transmitidos y entendidos en diferentes sistemas y dispositivos. Debido a que varias plataformas de computadoras almacenan información de manera diferente, ASN.1 suministra un formato común. Por ejemplo, un entero en un sistema puede ser expresado con el dígito más significativo primero, mientras que en otros podría ser expresado con el dígito menos significativo primero. Además, los tipos de datos ASN.1 se encuentran en las MIB. Para poder entender la administración de red, es necesario poder leer y entender objetos MIB. Sin un buen dominio de las MIB, será difícil usar las herramientas y el software SNMP para administrar de manera efectiva una red.

Se definen dos tipos de objeto de datos para usar ASN.1:

- tipos universales
- tipos de aplicación

Tabla 2.1 Tipos de datos universales.

Tipo de datos ASN.1	Descripción
INTEGER	Es un tipo de dato que representa un número cardinal, donde no se plantea ninguna limitación sobre el nivel de precisión que se requiere para representar un valor entero
OCTET STRING	Es un tipo de dato que representa cero o más octetos, donde cada octeto puede tomar cualquier valor desde 0 hasta 255
NULL	Es un tipo de dato que pretendía ser un sujetador (*placeholder*), pero que actualmente no se usa

Tabla 2.1 Tipos de datos universales. (*continuación*)

Tipo de datos ASN.1	Descripción
`OBJECT IDENTIFIER`	Es un tipo de dato que representa un objeto nombrado autoritariamente y que está formado por una secuencia de valores que especifican un árbol MIB
`SEQUENCE` `SEQUENCE OF`	Es un tipo de dato usado para denotar una lista ordenada de cero a más elementos que contienen otros tipos de ASN.1. `SEQUENCE OF` contiene una lista ordenada del mismo tipo ASN.1

La clase universal está conformada por tipos primitivos que pueden ser usados para construir tipos de datos adicionales de objetos en una MIB. La tabla 2.1 enumera algunos de los tipos de datos simples que se encuentran en la clase universal.

Tabla 2.2 Tipos de datos de aplicación.

Tipo de datos ASN.1	Descripción
`Counter` `Counter32`	Es un tipo de dato que representa enteros no negativos, que aumentan hasta alcanzar un valor máximo y luego se devuelven hasta cero. Un `counter` es un `INTEGER` que puede tomar un valor entre `0` y `4294967295`. Un `counter` no tiene un valor de inicio definido
`Counter64`	Es lo mismo que un objeto `counter`, excepto que el `counter64` es un `INTEGER` que puede tomar un valor entre `0` y `18446744073709551615`

Tabla 2.2 Tipos de datos de aplicación. (*continuación*)

Tipo de datos ASN.1	Descripción
`DisplayString`	Es un tipo de dato que representa cero o más octetos, donde cada octeto puede tomar cualquier valor entre `0` a `255`. Un `DisplayString` es como un objeto `OctectString`
`Gauge` `Gauge32`	Es un tipo de dato que representa un entero no negativo que puede aumentar o disminuir y se disparará a su máximo valor. Un `gauge` es como un `counter` en todos los otros aspectos
`IpAddress`	Representa un `OCTET STRING` que tiene una longitud de 4 bytes (32-bits) y donde cada uno de los cuatro octetos se relaciona con los cuatro bytes de una dirección estándar IP
`Opaque`	Es un tipo de dato que suministra la habilidad de pasar información arbitraria que usa el tipo de datos `OCTET STRING`
`NetworkAddress`	Puede representar una de varias direcciones estándar de protocolo de redes. Actualmente es lo mismo que `IpAddress`
`TimeTicks`	Representa un número entero no negativo que cuenta el tiempo en centésimas de segundo, desde una época establecida. `TimeTicks` es como un `counter` en todos los otros aspectos

SNMP obtiene algunos tipos de datos de aplicaciones de clase universal. Estos tipos de aplicación definen tipos adicionales que se pueden usar para representar valores específicos personalizados, para el uso dentro del ambiente de administración de red. La tabla 2.2 describe algunos de los tipos de datos de aplicaciones disponibles en el presente en la clase de aplicación.

¿Por qué incluir estos detalles de las MIB en este libro? Como las secciones de los capítulos siguientes se enfocarán en el uso de herramientas de administración de red que manipulan los objetos MIB, parece razonable presentar los tipos de objetos que puedan encontrarse. Por tanto, es importante que el lector entienda los diferentes tipos de objetos y por qué los valores de estos objetos están en un formato y estructura determinados. De igual importancia es el modo de acceso de los objetos MIB, el cual es analizado más adelante.

Ejemplos de objetos MIB

Debido a que los objetos MIB son un componente importante de la administración de red, algunos de los capítulos de este libro describen específicamente definiciones MIB, en la medida en que se relacionan con los agentes del sistema y con las herramientas de la administración de red. Por tanto, cuando se hable de las MIB se utilizará el siguiente formato:

Nombre de objeto: `sysDescr`
OID: `system.1`
Tipo de objeto: `Octet String`
Modo de acceso: `sólo lectura`
Estado: `actual`
Descripción: descripción del dispositivo o entidad agente. Este valor debería incluir el nombre completo, identificación del sistema, tipo de hardware, sistema operativo y software de la red. Es obligatorio que ésta contenga únicamente caracteres imprimibles ASCII. Un ejemplo de este objeto incluye lo siguiente, obtenido para el agente del sistema Linux: `Linux socrates 2.2.5-15 apr 19 22:21:09 EDT 1999`

Este formato incluye el *nombre del objeto*, la *cadena OID*, el *tipo del objeto*, el *modo de acceso*, el *estado* y la *descripción*. El nombre del objeto es el nombre usado cuando se consulta a un agente para este objeto particular. En este ejemplo, el objeto `sysDescr` es una cadena que contiene una descripción general del agente o dispositivo contenido en el dispositivo. La cadena del *identificador de objeto*, o cadena OID, muestra en qué grupo el objeto está contenido, y su posición lógica en esa jerarquía MIB.

En este caso, está en el primer objeto, en el grupo `system`. Este grupo puede encontrarse en la definición estándar MIB-II.

El *tipo del objeto* es `OctetString` y puede tener una longitud de 255 caracteres. Recuérdese que la tabla 2.2 enumera las definiciones de estos tipos ASN.1. El *modo de acceso* indica cómo el administrador u otras herramientas pueden manipular el objeto. Es muy común que personas sin experiencia en SNMP intenten alterar objetos no escribibles. En este caso, intentar alterar el objeto `sysDescr` causará que el agente responda con `not writable` u otro mensaje de error.

El campo de estado indica si el objeto es `current` (actual) o `historic` (histórico). El estado `current` significa que el objeto está disponible en el agente, como se describe en la MIB. Es decir, si la MIB representa a ese agente, entonces todos los objetos etiquetados como actuales se implementarán en el agente. Otros valores también pueden ser utilizados para indicar estados adicionales de los objetos. Por ejemplo, la referencia `obsolete` indica que el objeto no es soportado más en la MIB. La referencia depreciado (`deprecated`) puede usarse para ayudar a la interoperabilidad con versiones o implementaciones anteriores del agente.

El campo descripción (*description field*) suministra una lista general del objeto, y propósito, y un valor de ejemplo, si es del caso. En el ejemplo anterior, cuando se consulta el objeto `sysDescr`, éste retorna el sistema operativo del agente (`Linux`), el nombre del *host* donde reside el agente (`socrates`), la información de la versión del sistema operativo y la información de la fecha.

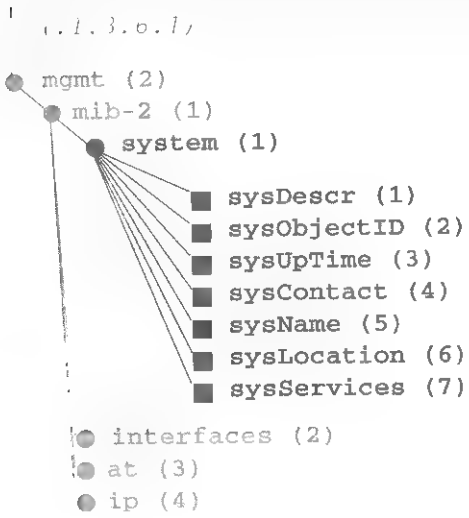


Figura 2.4 Ejemplo de estructura MIB.

Otra forma en que este libro describe las MIB es mediante una gráfica jerárquica que muestra cada uno de los objetos establecidos en la estructura del árbol. La figura 2.4 muestra el formato general. En particular, muestra los objetos MIB-II encontrados bajo el grupo `system`. Como se puede observar, el formato incluye el uso de círculos rellenos (●) y de cuadrados rellenos (■). Los cuadrados representan objetos discretos como `INTEGER` y `OCTET STRING`, pero no tablas u otros grupos de objetos. Los círculos representan grupos de objetos o tablas. Además, las líneas grises y objetos que son periféricos indican la estructura necesaria para navegar al grupo en cuestión. Estos objetos y líneas asociadas que son de interés aparecen en negro. Se incluye con estos objetos el índice del miembro (entre paréntesis) que muestra la posición relativa de cada objeto dentro de la estructura de árbol. Así, `sysContact(4)` es el cuarto objeto del grupo `system`.

Tablas

Como se anotó anteriormente, una MIB puede contener objetos que representan características físicas de un dispositivo u otra información contenida en el agente. Estos objetos pueden estar en forma de elementos discretos (por ejemplo, elementos individuales como `sysDescr`) o, en ciertos casos, como tablas bidimensionales. Las tablas almacenan información relacionada que puede contener varias instancias o copias de un objeto MIB. La mejor forma de ilustrar el uso de una tabla es examinándola en una MIB real.

Definido en el estándar MIB-II, se encuentra el grupo `interface`, que tiene definido al identificador de objeto como `1.3.6.1.2.1.2` o `iso.org.dod.internet.mgmt.mib-2.interface`. Los objetos de este grupo representan interfaces físicas de red e información relacionada instalada en un dispositivo de red. La información relacionada con el rendimiento es también recolectada y almacenada dentro de este grupo.

Para cada interfaz en un dispositivo de red, se utiliza la siguiente información para describir la naturaleza de la interfaz y su configuración asociada.

- *Descripción*: descripción general de la interfaz
- *Tipo*: el tipo de interfaz, como Ethernet o TokenRing
- *Mtu*: tamaño máximo de transmisión
- *Velocidad*: velocidad de transmisión de la interfaz

- *Dirección física*: protocolo de enlace de datos o dirección de hardware
- *Estado de administración*: estado administrativo actual de la interfaz
- *Estado operacional*: actual estado operativo de la interfaz
- *Último cambio*: tiempo en que la interfaz se volvió operacional

Objetos adicionales dentro de la tabla almacenan la siguiente información de seguimiento del rendimiento:

- Número de octetos recibidos o enviados
- Número de paquetes de unidifusión (*unicast*) entregados a un software de alto nivel, o enviados desde éste
- Número de paquetes diferentes de unidifusión entregados a un software de más alto nivel, o enviados desde éste
- Número de paquetes de entrada/salida descartados
- Número de paquetes de entrada/salida que contienen errores
- Número de paquetes de entrada/salida descartados, debido a un mal protocolo
- Longitud de la cola del paquete de salida

La figura 1.5 muestra la estructura del grupo interface. Las tablas contienen información de la interfaz porque los dispositivos de red pueden contener muchas interfaces. Por ejemplo, un dispositivo enrutador o conmutador puede contener literalmente docenas de interfaces y a menudo soportar diferentes protocolos de red, como Ethernet, ATM o FDDI. El uso de una tabla proporciona una forma sencilla y conveniente de tener acceso a los objetos individuales, dada una definición de interfaz.

El grupo `interface` incluye el objeto `ifNumber`, el cual contiene el número total de interfaces en el dispositivo de red. Al usar el ejemplo de datos contenido en la tabla 2.3, el valor `ifNumber` sería `2`. En este caso, el dispositivo reporta dos interfaces: una definida como pseudointerfaz y la otra como una Ethernet. Es común que los dispositivos de red contengan una pseudointerfaz para fines de diagnóstico interno.

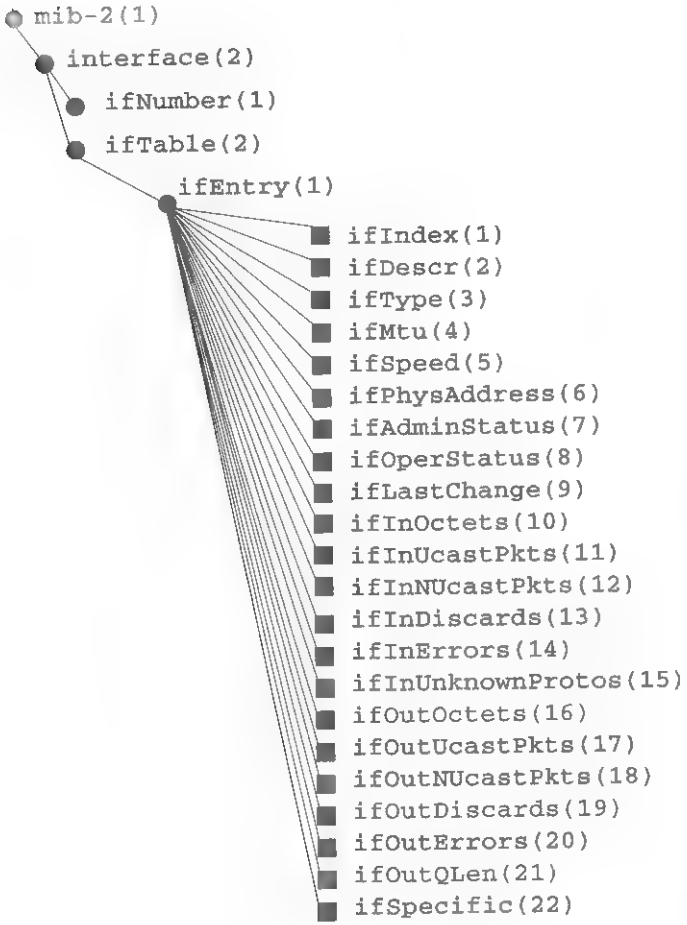


Figura 2.5 Estructura del grupo interface MIB-II.

Tabla 2.3 Tabla de instancias del grupo interface.

Objetos MIB	IfEntry 1.3.6.1.2.1.2.2.1	IfEntry 1.3.6.1.2.1.2.2.2
ifIndex	1	2
ifDescr	Pseudo Interface	Ethernet
ifType	1	6
ifMtu	1500	1500

Tabla 2.3 Tabla de instancias del grupo interface. (*continuación*)

Objetos MIB	IfEntry 1.3.6.1.2.1.2.2.1	IfEntry 1.3.6.1.2.1.2.2.2
ifSpeed	10000000	10000000
ifPhysAddress	None[a]	0x00040010ee5d
ifAdminStatus	1	1
ifOperStatus	1	1
ifLastCnange	0	0
ifInOctects	0	42617
ifInUcastPkts	445	680
ifInNUcastPkts	0	19
ifInDiscards	0	0
ifInErrors	0	5
ifInUnknown -Protos	0	0
ifOutOctets	0	42600
ifOutUcastPkts	445	570
ifOutNUcastPkts	0	94
ifOutDiscards	0	0
ifOutErrors	0	87
ifOutQLen	0	0
ifSpecific	null	null

a. Normalmente, las pseudointerfaces no tienen direcciones físicas asociadas con ellas. Es por eso que no aparece ninguna dirección de interfaz en esta lista.

El resto del grupo `interface` consiste en una tabla llamada `ifTable`, la cual contiene una fila para cada interfaz definida en el dispositivo. Esta tabla está indexada por `ifIndex`, el cual contiene un valor entre el rango de 1 y el valor de `ifNumber`. El objeto `ifIndex` únicamente señala cada columna de la tabla que representa la interfaz.

La tabla 2.3 muestra todo el grupo `interface` en la primera columna y los valores asociados tomados de un dispositivo real de red en las columnas restantes. Cada fila corresponde a cada uno de los objetos MIB específicos contenidos en la tabla `ifEntry`. Cada instancia `ifEntry` representa una interfaz definida en la tabla. Para acceder al objeto `ifDescr` para la primera interfaz, uno debería usar el identificador de objeto (OID):

```
1.3.6.1.2.1.2.2.1.1
```

o

```
iso.org.dod.internet.mgmt.mib-2.interface.ifTable.ifEntry.ifDescr
```

El objeto `ifDescr` proporciona una descripción general de la interfaz. Los objetos, empezando desde `ifInOctets` hasta `ifOutQLen`, representan contadores de tráfico en el agente para interfaces de dispositivos y pueden usarse para medir el rendimiento de la red y de los dispositivos.

Acceso a objetos

Los objetos MIB están definidos con información de control de acceso que especifica qué tipo de operación puede ejecutarse en el objeto. El SNMP incluye la siguiente información de control de acceso a los objetos MIB:

- no accesible (*not-accessible*)
- accesible por notificación (*accessible-for-notify*)
- sólo lectura (*read-only*)
- lectura-escritura (*read-write*)
- lectura-creación (*read-create*)

Los objetos *not-accessible* (no accesibles) definidos en la MIB usualmente referencian definiciones de objeto u otras descripciones de objetos que no son objetos en sí mismos, que pueden ser manipulados por un administrador SNMP. Un buen ejemplo es

la estructura de datos de datos de la tabla, donde un objeto describe la forma o el tamaño de la tabla pero no las filas y columnas reales.

Los objetos *accessible-for-notify* (accesible por notificación) están disponibles solamente por medio de una notificación a un administrador de red u otro agente. No se permite el sondeo directo del objeto.

Los objetos *read-only* (de sólo lectura) no son alterables por el sistema de administración de red, pero los valores pueden ser obtenidos por medio de una operación get o trap. ¿Por qué estaría prohibido hacer un cambio en un objeto MIB? La razón es muy clara: cierta información de la MIB nunca cambiará durante la vida del producto. Por ejemplo, el objeto MIB `sysDescr`, el cual quiere decir descripción del sistema (*system description*), contiene información del vendedor para el agente. Un administrador SNMP no debe modificar esta información porque separaría al dispositivo del vendedor real del producto, haciendo difícil así la identificación del agente. Además, puede afectarse adversamente la exactitud de cualquiera de los mecanismos de inventario de red basados en el software. Otra razón para hacer objetos de sólo lectura es asegurarse de que la información de rendimiento u otros datos estadísticos permanezcan exactos, en lugar de que sean alterados no deliberadamente.

El acceso *read-write* (de lectura-escritura) es necesario cuando debe alterarse un objeto particular para cumplir una meta específica, o debe configurarse de cierta manera. Por ejemplo, puede ser necesario inhabilitar un puerto de un enrutador, debido a un gran número de errores detectados en una de sus interfaces. En este caso, el sistema de administración de red puede cambiar el estado de operación de la interfaz a `0`, apagando así la conexión física hasta determinar la causa del error.

Los objetos *read-create* (de lectura-creación) tienen el mismo permiso de acceso que los objetos de sólo lectura y lectura-escritura. El acceso de lectura-creación se usa para objetos que deben ser creados sobre la marcha. Tales objetos pueden incluir instancias de filas de tablas (también llamadas filas conceptuales), por ejemplo.

MIB estándar y privadas

Como se mencionó anteriormente, las MIB están organizadas bajo una estructura jerárquica de árbol, y varias MIB de estándar se han desarrollado y colocado en la rama `mgmt(2)`.

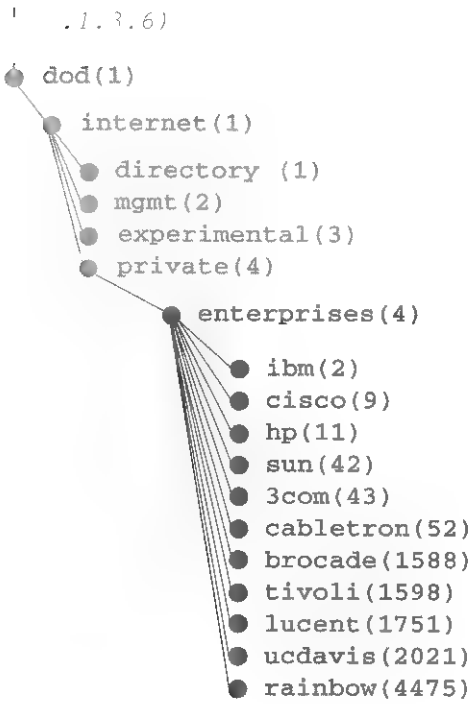


Figura 2.6 MIB empresariales de una rama privada.

Muchas de estas MIB fueron desarrolladas a través del proceso RFC. Muchos vendedores o individuos colaboraron en su desarrollo, pero en realidad no son sus propietarios, ni pueden realizar

Tabla 2.4 Números asignados a empresas.

Número	Vendedor
2	IBM
9	Cisco Systems
11	Hewlett-Packard
42	Sun Microsystems
43	3Com Corporation
52	Cabletron Systems
1588	Brocade Communications Systems

Tabla 2.4 Números asignados a empresas. (*continuación*)

Número	Vendedor
1598	Tivoli Systems
1751/3729	Lucent Technologies
4475	Rainbow Software Solutions

cambios en ellas arbitrariamente. Sin embargo, muchos vendedo res y desarrolladores de software de terceros han desarrollado MIB adicionales para que aborden una funcionalidad específica o servicios para sus productos particulares. Muchas de estas MIB, conocidas como MIB empresariales, inician bajo la rama `private(4)` del árbol MIB estándar. La figura 2.6 muestra varias MIB populares que pueden controlarse bajo esta rama.

La Autoridad de Números Asignados en Internet, *Internet Assigned Numbers Authority* (IANA), mantiene una lista de números asignados a las empresas. Quienes desean obtener uno de éstos, hacen una solicitud a esta organización y obtienen un número válido. La tabla 2.4 muestra un pequeño ejemplo de los números que ya se han asignado.

¿Por qué desarrollar MIB empresariales? Muchos vendedores lo hacen porque necesitan objetos específicos para sus dispositivos que pueden no estar definidos o disponibles en las MIB estándar. Además, las MIB específicas de vendedores no necesitan convertirse en estándar, ya que sólo dicho vendedor y sus clientes necesitan utilizar realmente las MIB. Desarrollar MIB estándar sólo tiene sentido cuando los vendedores pueden implementar la misma colección de MIB a través de un gran número de dispositivos diferentes.

Comunidades SNMP

A menudo, los sistemas de administración de redes serán instalados en un gran entorno de red que contiene una colección de muchos grupos diferentes de redes y dispositivos. Por tanto, es razonable que una red empresarial se divida en zonas o comunidades de usuarios, para dividir responsabilidades. Como resultado, un nombre de comunidad (*community name*) puede ser asignado a una clase de dispositivos, y suministrar un límite de seguridad que ayude a implementar las zonas o comunidades descadas. El SNMP apoya este tipo de modelo de seguridad basado en la información de la cadena de la comunidad, la cual

está añadida físicamente a cada uno de los dispositivos en la comunidad seleccionada. Algunos ejemplos prácticos incluyen la selección de una comunidad que represente todos los enrutadores vertebrales de Cisco y la selección de otra comunidad que incluya dispositivos sólo del departamento de ventas.

El modelo de autenticación actual del SNMP basado en comunidades es considerado muy débil y plantea un problema significativo de seguridad. La razón principal es que el SNMP no proporciona ninguna facilidad de encriptación u otros mecanismos que aseguren que la información de las comunidades no sea simplemente copiada de la red durante un intercambio de paquetes SNMP. Usando una herramienta de captura de paquete, todo el paquete SNMP puede ser decodificado, revelando así el nombre de la comunidad. Como resultado de esta limitación, muchos sitios inhabilitan las operaciones *set* para muchos de los dispositivos del agente. Esto tiene un desafortunado efecto, al limitar la utilidad del SNMP, porque éste sólo puede hacer un seguimiento de los valores de los objetos, pero no alterarlos.

Vista general de las versiones de SNMP

El SNMP llegó a estar completamente disponible en 1988. Desde entonces, su uso se ha extendido y ha sido implementado en casi todos los dispositivos de redes de área local (LAN) y en muchos sistemas operativos de computadora. Esta versión popular, conocida como SNMPv1, hoy en día explica una porción significativa de la base instalada de agentes. A pesar de su popularidad, la SNMPv1 tiene varios problemas fundamentales. Primero, debido a que carece de un mecanismo robusto de seguridad, no puede ser utilizada con todo su potencial. Como resultado, muchos vendedores limitan las operaciones **set** en los agentes, para minimizar el riesgo potencial de una violación de seguridad. Segundo, SNMPv1 no optimiza el procesamiento de grandes cantidades de información; por tanto, restringe además el uso del SNMP. Tercero, la relación de la SNMPv1 entre el administrador de red y el agente está bien definida; los agentes desempeñan sólo el único y sencillo papel de aceptar comandos desde otros sistemas de administración. Esto limita significativamente a la SNMPv1 cuando se necesitan agentes inteligentes, con el fin de abordar requerimientos específicos para funciones distribuidas de la administración de red. Además, la SNMPv1 especifica una colección de

objetos MIB conocida como MIB-II. El objetivo de la MIB-II era proporcionar una colección de objetos que todos los agentes de SNMP puedan soportar, sin importar qué vendedor implementó el agente. En resumen, la SNMPv1 proporciona:

- el marco de trabajo básico de la administración de red
- la definición de la MIB-II estándar
- las descripciones de las unidades de datos de protocolo (PDU), las cuales incluyen *GetRequest, GetNextRequest, SetRequest, GetResponse y Trap*
- una descripción del lenguaje de codificación ASN.1

A fin de subsanar algunas de las deficiencias en la SNMPv1, en los últimos años se han hecho esfuerzos significativos para mejorar el SNMP. La primera serie de mejoramientos se presentó en 1993, cuando fue introducida una serie de 12 RFC (1441-1452), proponiendo añadir PDU y ampliaciones a la arquitectura del SNMP y al modelo de seguridad. Esta colección de RFC fue conocida como SNMPv2 Classic. En ese momento, muchos usuarios del SNMP se estaban anticipando a estas mejoras y planearon implementar la nueva versión tan pronto ésta estuviera disponible.

Desafortunadamente, había aún mucha discusión acerca de la seguridad de la SNMPv2 y la configuración remota. Cuando se hizo evidente que no todas las propuestas originales de la SNMPv2 iban a ser adoptadas ampliamente, se realizó trabajo adicional para definir la seguridad del SNMP y de los componentes remotos de administración de configuración. Esto llevó a propuestas adicionales que incluían un nuevo protocolo y nuevos elementos de la estructura MIB (documentados en las RFC 1901-1908). Estas propuestas fueron más populares que las de la SNMPv2 Classic y fueron conocidas como base de la comunidad SNMPv2 o SNMPv2c. El problema con la SNMPv2c era que, aunque fue respaldada por la IETF, carecía de una seguridad robusta y de facilidades de administración.

Mejoramientos adicionales para la capacidad de administración remota del SNMPv1 dieron como resultado propuestas conocidas como la SNMPv2usec y SNMPv2*, documentadas en la RFC 1909-1910 y en la RFC 2222, respectivamente. La SNMPv2usec recomienda un modelo de seguridad robusto y un

marco de trabajo administrativo. Uno de los problemas con la SNMPv2usec es que carece de respaldo de la IETF, la cual abandonó la propuesta por una no estándar. Las funciones principales de la SNMPv2 incluyen:

- tipos de datos expandidos, como los contadores de 64-bit
- mejoramiento del desempeño y de la eficiencia con la PDU *get-bulk*
- notificación de evento con la PDU *inform*
- mejor manejo de error
- creación y borrado más eficiente de registro

Por último, y como un intento para alcanzar algún acuerdo entre las variadas propuestas de administración remota y de seguridad, se escribió otra serie de RFC, la cual posteriormente vino a ser conocida como la SNMPv3. Estas RFC (2271 2275) fueron formuladas por la IETF como normas estándar, lo que significa que están disponibles para el público en general, para su revisión y análisis. La SNMPv3 es la SNMPv2c más las provisiones para una seguridad y una administración robustas; ésta utiliza las SNMPv2c RFC (1902-1908). En un alto nivel, la propuesta SNMPv3 intenta:

- proponer un modelo de seguridad basado en DES, MD5 y otros protocolos de autenticación
- definir un modelo de control de acceso basado en vistas
- redefinir algunos de los conceptos y términos del SNMP

La SNMPv3 ha disfrutado en este punto de cierto nivel de éxito en el mercado, pero los protocolos siguen siendo muy nuevos. Las principales compañías de software y hardware de trabajo en red ya soportan la SNMPv3 o están trabajando en ella en la actualidad. Sin embargo, la SNMPv3 todavía tiene que ser instalada ampliamente en la industria; esto es tal vez sólo cuestión de tiempo. Uno de los retrasos principales para la adopción de este nuevo protocolo es la proporción costo-beneficio. Aquellos sitios que requieren hoy en día una mayor seguridad, se pueden actualizar con los nuevos protocolos tan pronto como los vendedores pongan a disposición su implementación. Sin embargo, no se justifica que todos actualicen todo el equipo disponible

para soportar la SNMPv3. El enfoque más probable puede ser migrar lentamente a través del tiempo, a medida que se instalan los productos más nuevos que contienen soporte SNMPv3. Finalmente, los nuevos productos remplazarán a los antiguos, e instalar los protocolos nuevos de administración será sólo un problema de configuración.

Operación de protocolo SNMP

Un mensaje SNMPv1 contiene tres componentes principales: un campo de versión, un campo de comunidad y un campo unidad de datos de protocolo SNMP (PDU). A diferencia de otros protocolos TCP/IP, los paquetes SNMP no son de una longitud fija; en cambio, cuentan con un formato ASN.1. La figura 2.7 muestra la estructura básica del paquete SNMP y una descripción de los campos (y tamaños).

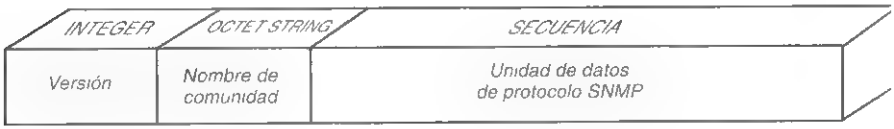


Figura 2.7 Formato de mensaje SNMP.

El formato incluye nombres de campo, mostrados en **negrita**, y para las etiquetas de formato (tipo), como se muestra en la parte superior en texto gris. El tipo muestra el formato básico para cada uno de los campos. Por ejemplo, la información de la versión es un objeto `INTEGER`. Los nombres de campo son definidos como sigue:

versión: este campo indica cuál versión del protocolo SNMP se está usando. En la actualidad, la versión 1 es el protocolo SNMP más ampliamente implementado y soportado.

nombre de comunidad: la comunidad se usa como el mecanismo de seguridad primario para establecer la autenticación desde un administrador de red hasta los agentes en el SNMP. El nombre de comunidad o cadena se usa como una palabra clave (*password*) cuyas solicitudes para acceder a objetos son permitidas, con base en la condición de que el administrador de red conozca la

palabra clave del agente. Si el agente ha sido configurado para emitir trampas (*traps*), una trampa `authenticationFailure` se genera cuando un administrador de red consulta un agente con una cadena de comunidad inválida.

unidades de datos de protocolo:

las PDU SNMPv1 pueden ser uno de cinco diferentes tipos, y constan de componentes de solicitud y respuesta. Éstos incluyen:

- *GetRequest*
- *GetNextRequest*
- *SetRequest*
- *GetResponse*
- *Trap*

La SNMPv2 define las siguientes PDU adicionales:

- *GetBulkRequest*
- *InformRequest*

Cada uno de los componentes GetRequest, GetNextRequest y SetRequest obtiene del agente de respuesta una GetResponse que puede contener datos válidos o un estado de error.

GetRequest

El administrador o aplicación SNMP emite una PDU GetRequest, para obtener uno o más objetos MIB de un agente SNMP. La figura 2.8 muestra el formato del paquete que contiene los siguientes campos:

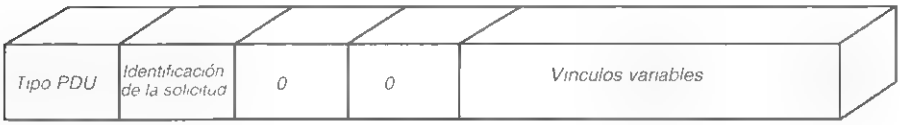


Figura 2.8 Formato de mensaje GetRequest.

tipo pdu:	indica que el tipo PDU es un GetRequest.
identificación de la solicitud (request-id):	identificador único que permite al administrador del SNMP emparejar solicitudes y respuestas; también ayuda a la detección de mensajes duplicados que pueden ser recibidos cuando se usa un servicio de transporte poco confiable.
vínculos variables:	es una lista de objetos MIB solicitados.

La operación GetRequest es la forma principal de obtener información de los agentes cuando los objetos en cuestión son conocidos de antemano. Por ejemplo, si el administrador de red decidiera recuperar los objetos `sysDescr` y `sysUpTime` de un agente, podemos pensar en la solicitud como una simple función que incluye:

```
GetRequest (sysDescr,sysUptime)
```

En este caso, estos dos objetos están ubicados dentro del campo del vínculo variable, cuando éste se envía. El objeto `sysDescr` representa una cadena que contiene una descripción general del agente, y `sysUpTime` refleja la cantidad de tiempo que un agente ha estado corriendo.

Cuando el agente recibe el mensaje anterior y no se presenta error, responderá con los valores de los objetos MIB solicitados por medio de una PDU GetResponse. La operación GetRequest es atómica. Es decir, o retornan todos los valores solicitados o no retorna ninguno. Cuando la entidad receptora responde al GetRequest, incluye los valores en el campo del vínculo variable GetResponse. Si, por alguna razón, por lo menos uno de los valores no puede ser suministrado, no se proporcionará ningún valor.

Si los valores solicitados desde un administrador no pueden ser retornados por el agente, el agente responderá con un error. Por ejemplo, el sistema de administración puede haber solicitado un objeto MIB que no esté implementado en el agente. En este caso, el agente posiblemente no puede satisfacer la solicitud y, por tanto, devuelve un error. Además, en ciertas situaciones, retornar el valor de un objeto solicitado falla porque éste puede ser demasiado grande para ser procesado por el administrador.

El campo de *vínculos variables* incluye una lista de objetos cuyos valores son solicitados. Cuando el agente responde a través

de GetResponse, el vínculo variable incluye los objetos solicitados y los valores asociados. Dado el ejemplo anterior, el siguiente GetResponse será enviado por el agente:

```
GetResponse(SysDescr-"3com Enterprise Monotor",SysUptime\
-0000154477)
```

En este ejemplo, el agente es un seguidor RMON de 3Com Enterprise (o sonda de red), como se indica en el campo `sysDescr`. Este dispositivo recolecta información del rendimiento de red e información de contenidos de paquetes para un análisis y reporte posteriores. El objeto `sysUpTime` contiene la cantidad de tiempo que el agente ha estado arriba (*up*) y corriendo. El valor mostrado en el ejemplo indica que el dispositivo estuvo corriendo aproximadamente durante 15 minutos y 44 segundos.

GetNextRequest

La PDU GetNextRequest es similar a la PDU GetRequest, y los formatos de paquete son idénticos, como lo muestra la figura 2.8. Sin embargo, la PDU GetNextRequest tiene una diferencia: se usa para recuperar objetos cuando se desconoce la estructura del árbol MIB. La GetNextRequest puede ser una gran ventaja cuando se utiliza para descubrir la estructura MIB exacta de un agente. ¿Por qué la estructura MIB de un agente sería desconocida por un administrador SNMP? El SNMP proporciona una forma indirecta para determinar cuáles MIB o estructura MIB son soportadas en un agente. Los vendedores tienen la libertad, por supuesto, para implementar cualquiera de las MIB que necesiten sus dispositivos. Por consiguiente, el administrador de red debe descubrir las MIB soportadas, recorriendo el árbol MIB de forma automatizada. Cuando el GetNextRequest es enviado con un objeto particular, el GetResponse retorna el valor del objeto solicitado, más la instancia del siguiente objeto lexicográfico en el árbol MIB. Como resultado, cada GetNextRequest revelará el siguiente objeto en la MIB, sin que el administrador conozca cuál será el siguiente objeto. La operación GetNextRequest también suministra un mecanismo más eficiente para recuperar objetos de un agente que GetRequest, porque éste requiere menos intercambios solicitud/respuesta.

SetRequest

La operación SetRequest se emplea para alterar información del agente, como el valor de un objeto MIB. El formato del paquete

es el mismo que GetRequest y GetNextRequest, como se muestra en la figura 2.8. A diferencia de GetRequest o GetNextRequest, SetRequest se usa para alterar el valor de un objeto MIB. Como se mencionó anteriormente, la operación SetRequest requiere privilegios de seguridad, los cuales son transformados actualmente a través de la cadena de comunidad para el protocolo SNMPv1. Por tanto, el agente debe validar la cadena de comunidad SetRequest suministrada antes de que a la operación se le permita tener éxito. La operación SetRequest es también atómica; o todos los valores pueden ser alterados, o ninguno puede serlo. Si la operación SetRequest es exitosa, una GetResponse es devuelta, y la lista de vínculo variable contiene los objetos con sus nuevos valores. Así es como el administrador de red puede determinar que la operación set fue exitosa. Si SetRequest no tuvo éxito, un error es devuelto en GetResponse

GetResponse

Cada una de las operaciones SNMP, con la excepción de la trap, recibe una GetResponse del agente. La figura 2.9 muestra el formato del paquete GetResponse, el cual incluye los siguientes campos:

tipo pdu:	indica el tipo PDU, el cual es GetResponse.
identificación de la solicitud:	único identificador que permite el emparejamiento de solicitudes y respuestas.
estado de error:	indica que ocurrió una condición de excepción durante el procesamiento de la solicitud.
índice de error:	cuando ocurre un error, indica cuál objeto variable en la lista de vínculo variable causó el error.

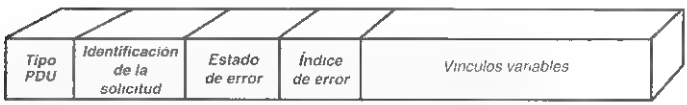


Figura 2.9 Formato de mensaje GetResponse.

vínculos variables: una lista de objetos MIB que están involucrados en la operación.

Trap

Una *Trap* es un mensaje no solicitado, dirigido desde un agente hasta una estación de administración de red, que representa un evento o una condición significativa cuya notificación al administrador se considera necesaria. Este tipo de comunicación desde el agente es asincrónica, comparada con el sondeo desde el administrador. La PDU Trap es muy diferente de las otras PDU definidas en el SNMP, como se muestra en la figura 2.10. A diferencia de los otros PDU SNMP, la Trap no garantiza una respuesta de la red receptora. Los campos de Trap incluyen:

Tipo PDU: indica que el tipo PDU es una Trap.

Empresa: contiene el objeto `sysObjectID` MIB del agente emisor. El objeto `sysObjectID` incluye información respecto al vendedor del agente que envió la Trap.

Dirección del agente: representa la dirección IP del agente emisor.

Trap genérico: uno de los valores predefinidos de Trap enumerados en la tabla 2.5.

Trap específico: más información detallada acerca de la Trap. Ésta es usualmente `zero`, a menos que la Trap genérica sea una Trap `enterpriseSpecific`. Las Traps empresariales son específicas del vendedor y pueden contener información adicional del agente acerca de la condición que causó la Trap en el primer lugar.

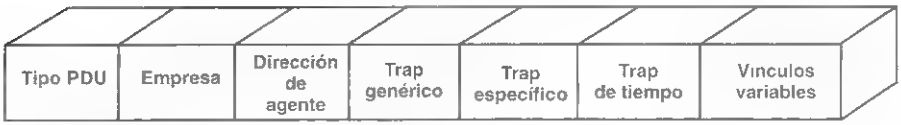


Figura 2.10 Formato de mensaje Trap.

Sello de tiempo: (*Time-stamp*) es la cantidad de tiempo entre la generación de la Trap y el momento en que el dispositivo fue iniciado por última vez, expresada en décimas de milisegundos.

Vínculo variable: información específica del vendedor relacionada con la Trap.

Tabla 2.5 Trampas predefinidas de SNMP.

Tipo de Trap	Significado
`ColdStart (0)`	El dispositivo está arrancando de nuevo o reinicia por sí mismo, de modo que el agente o la configuración puede cambiarse. Usualmente, esto indica una caída o cualquier otra condición de reinicio
`WarmStart (1)`	El dispositivo está arrancando de nuevo o reinicia por sí mismo, de tal forma que no se hacen cambios al agente o configuración. Usualmente esto implica un simple *refresco* o reinicio del sistema operativo
`LinkDown (2)`	Indica una falla en uno de los enlaces de comunicaciones de dispositivo (interfaz)
`LinkJp (3)`	Indica que el enlace de la comunicación de dispositivo (interfaz) está ahora arriba y corriendo

Tabla 2.5 Trampas predefinidas de SNMP. (*continuación*)

Tipo de Trap	Significado
`AuthenticationFailure (4)`	Ha ocurrido una falla de autentificación o seguridad en el dispositivo. Comúnmente, esto indica que se ha utilizado una cadena de comunidad SNMP inválida
`EgpNeignborLoss (5)`	Indica que el vecino del protocolo externo de puerta de acceso (*gateway*), cuyo dispositivo es un par, ha sido etiquetado abajo (*down*) y la relación ya no es válida
`EnterpriseSpecific (6)`	Indica que ha ocurrido algún evento específico del vendedor. Los vendedores usan este tipo de trampa genérica para representar sus propias trampas propietarias

La tabla 2.5 enumera los siete tipos predefinidos de Trap generales. El tipo de Trap `enterpriseSpecific` es suministrado como un mecanismo para definir trampas personalizadas o propietarias que no se clasifican entre los otros tipos genéricos.

GetBulkRequest

El administrador o la aplicación SNMPv2 emite esta PDU para minimizar la interacción de la red y permitir al agente retornar paquetes más grandes (en comparación con GetNextRequest y GetRequest); por tanto, mejora la eficiencia al obtener un número mayor de objetos de un agente. Éste usa el mismo formato PDU como muchos de los otros operadores SNMPv1. La única diferencia es el renombramiento de los campos *estado de error* e *índice de error* (de una respuesta PDU) como *no repetidores* y *repeticiones máximas*, respectivamente. Estos campos se definen como sigue:

no repetidores:	es el número de objetos MIB que deberían ser recuperados como máximo una vez.
repeticiones máximas:	es el número máximo de veces que otros objetos MIB deberían ser recuperados.

InformRequest

La PDU InformRequest es emitida por una entidad SNMPv2 que actúa en el papel de administrador, en relación con otra entidad SNMPv2 que actúa en el mismo papel, con el propósito de suministrar información de administración de red. La función principal de esta PDU es suministrar capacidades distribuidas de administración SNMP. Por tanto, un agente puede utilizar esta PDU para suministrar servicios y funciones parecidas a las de la administración. El formato de esta PDU es el mismo que para GetRequest y otras PDU relacionadas.

Códigos de respuesta SNMP

Los códigos de error retornados desde un agente SNMPv1 eran muy limitados. Por ejemplo, si un administrador SNMP solicitaba la operación set en un objeto MIB, y el agente no podía realizar la operación como se solicitó, el agente respondía con `noSuchName`. Con la adición de más códigos de error en la SNMPv2, el agente respondería en esta situación con `notWritable`. La tabla 2.6 enumera todos los códigos de respuesta del SNMP.

Tabla 2.6 Códigos de respuesta SNMP.

Código de respuesta	Descripción
SNMPv1	
`tooBig`	Retornado por el agente si la respuesta a la solicitud fuese demasiado grande para enviarla
`noSuchName`	Retornado por el agente en cualquiera de estos dos casos: 1) si se intenta una operación **set**

Tabla 2.6 Códigos de respuesta SNMP. (*continuación*)

Código de respuesta	Descripción
	para un objeto que no está en la vista de la MIB, o 2) si se intenta una operación `set` para un objeto que está en la vista de la MIB, pero este objeto es de sólo lectura. Este código se usa ahora para la compatibilidad *proxy*
`badValue`	Retornado por el agente que ha detectado un error en la lista de vínculo variable PDU. Este código se usa ahora para la compatibilidad *proxy*
`Read-only`	Retornado por el agente. Este código se usa ahora para la compatibilidad *proxy*
`genError`	Retornado por el agente cuando el procesamiento de una PDU falla por una razón distinta a las enumeradas en esta tabla
SNMPv2/v3	
`noAccess`	La variable está fuera de la vista MIB definida para que esta operación tenga éxito
`notWritable`	La variable existe en el agente, pero el agente es incapaz de modificar el objeto
`WrongType`	El valor suministrado es del tipo de datos equivocado, como se define por la ASN.1
`WrongLength`	El valor suministrado es de longitud equivocada
`WrongEncoding`	El valor suministrado no fue codificado correctamente

Tabla 2.6 Códigos de respuesta SNMP. (*continuación*)

Código de respuesta	Descripción
`WrongValue`	El valor suministrado no está dentro del rango requerido para el tipo de objeto
`NoCreation`	El objeto no existe y el agente es incapaz de crear una instancia de este objeto
`InconsistentName`	El objeto no existe, y el agente es incapaz de crear una instancia de este objeto porque el nombre es inconsistente con los valores de otros objetos relacionados
`InconsistentValue`	El objeto suministrado es inconsistente con los valores de los objetos administrados
`resourceUnavailable`	Un recurso necesario en el agente no puede ser reservado para completar la solicitud

Estos códigos son importantes porque pueden ayudar a hacer un seguimiento de los problemas y de los temas cuando se usan agentes y herramientas SNMP que se comunican con los agentes. Conocer estos códigos de mensaje y entender la diferencia entre errores de configuración con el agente *versus* los problemas de conectividad entre el administrador y el agente, ayuda a determinar la solución para los problemas SNMP.

Transmisión de un mensaje SNMP

La siguiente serie de eventos ocurre cuando un administrador de red formula un mensaje SNMP:

1. Se constituye la PDU básica.
2. La PDU se pasa a la capa de servicio de seguridad si ésta se encuentra disponible.

3. La capa de protocolo formatea el mensaje, incluida la información de versión y comunidad.

4. El mensaje completo es ahora codificado usando las reglas ASN.1.

5. El mensaje se pasa al servicio de transporte, de modo que éste será entregado a la entidad receptora.

La siguiente serie de eventos ocurre cuando un dispositivo de agente recibe un mensaje SNMP:

1. Se realiza una verificación básica para asegurar que el mensaje es formateado correctamente. El mensaje se descarta si se encuentra cualquier error.

2. Se verifica la información de versión de protocolo. Si hay una falta de correlación, el mensaje es descartado.

3. El servicio de seguridad intenta verificar la entidad emisora. Si esto falla, se genera una trampa y el mensaje se descarta.

4. Se decodifica la PDU.

5. Se procesa la PDU.

Protocolo no orientado a la conexión

El SNMP es un protocolo no orientado a la conexión, lo cual significa que éste no soporta el concepto de establecer y controlar una conexión dedicada como Telnet o FTP. El SNMP transmite información entre un agente y un administrador mediante el uso de solicitudes y retorno de respuestas. Esto traslada el problema, de modo que los agentes no tienen que soportar protocolos adicionales y procesos asociados con los esquemas basados en la conexión. Por consiguiente, el SNMP proporciona su propio mecanismo para abordar la detección de errores y confiabilidad.

Agente maestro y subagente

Cuando se instala software de administración de red, es común tener un solo agente instalado en cada sistema que será administrado. En el caso de los dispositivos de trabajo en red como los enrutadores de red, y los conmutadores, usualmente sólo un agente está disponible. En cualquiera de los casos, los agentes se

comunicarán con uno o más de los administradores de red usando el puerto SNMP estándar 161. Para la mayoría de las necesidades de un solo agente tiene sentido y es apropiado. Sin embargo, hay momentos en que es necesario más de un agente para alcanzar el nivel deseado de manejo. Por ejemplo, los fabricantes de estaciones de trabajo suministrarán frecuentemente un agente SNMP con su sistema operativo. Este agente particular del sistema soportará usualmente un número limitado de funciones y parámetros del sistema operativo. Si se desea administrar los servicios de base de datos en este mismo sistema, a través del SNMP, esto requerirá la instalación de otro agente para que de manera específica haga un seguimiento de las funciones de base de datos. Esto plantea un problema porque el administrador SNMP tendrá acceso a los dos agentes usando el mismo puerto estándar. Además, cuando los agentes son arrancados por el sistema, el primer agente arrancará y abrirá el puerto 161 y operará de modo normal. Sin embargo, cuando el segundo agente arranque e intente hacer lo mismo, éste mostrará un error indicando que el puerto está ocupado y deberá abortar la operación.

Hay dos planteamientos posibles para abordar este problema de disputa de puerto. Primero, uno de los agentes puede ser configurado para usar un puerto alternativo, en lugar del estándar 161. Esta solución funcionará si el agente puede ser elaborado para soportar otro puerto cuando éste sea arrancado por el sistema. Muchos agentes de sistema soportan puertos alternativos, pero esta no es de ningún modo la regla. El principal inconveniente para esta solución es que cualquiera de los administradores SNMP usado para sondear al agente debe estar ahora configurado para usar también este nuevo puerto. Esto no plantea un problema significativo cuando se utiliza solamente un pequeño número de puertos no estándar. Sin embargo, si esta solución se utiliza a gran escala, digamos con muchos agentes que usan puertos no estándar a través de una gran cantidad de sistemas, el enfoque pierde su atractivo debido a la carga administrativa que puede generar.

La otra solución puede ser considerada más refinada porque involucra el uso de un agente maestro y de uno o más subagentes. Un agente maestro soluciona el problema del conflicto de puerto, convirtiéndose en el guarda del puerto estándar SNMP y enviando todo el tráfico SNMP al subagente apropiado. Los subagentes son agentes normales SNMP, pero están registrados con el agente maestro y asignados a puertos no estándar que se usan para comunicarse con el agente maestro. Los mensajes SNMP se envían desde los administradores hasta el agente maestro, que en

cambio entrega el mensaje al subagente correcto usando el puerto que éste asigna dinámicamente al subagente. La figura 2.11 muestra un diagrama de la arquitectura agente maestro-subagente. Un beneficio principal de esta solución es que alivia la tarea administrativa de modificación de los administradores SNMP citados en la primera solución. Además, esta solución puede ampliarse o mejorarse muy bien porque pueden ser añadidos nuevos subagentes de una manera sencilla.

Figura 2.11 Arquitectura agente maestro-subagente.

Del lado negativo, el agente maestro puede plantear un problema porque puede crearse o volverse inoperante. Esto tendría el efecto adverso de inhabilitar la conectividad con el subagente. Debido a que este problema en particular puede también ser similar en el caso de otros sistemas de software, el riesgo, aunque no es cero, no es necesariamente un factor significativo. Además, debido a que el agente maestro es responsable de entregar el SNMP al subagente apropiado, éste toma una cierta cantidad de tiempo de procesamiento y sobrecarga. Este puede ser un factor en las funciones de administración de tiempo crítico.

Se han presentado varias propuestas y productos, en un intento de suministrar un mecanismo estándar que soporte la arquitectura del agente maestro y del subagente. Éstos incluyen SMUX[1]

1. Este RFC ha sido promovido a estado histórico y, como tal, no puede recibir atención de los vendedores de productos de administración de red.

(RFC1227), AgentX (RFC2257), y productos específicos del vendedor, como el agente maestro en Solaris y el agente maestro en HP-UX. Todos éstos cumplen el mismo propósito: permiten a dos o más agentes operar en el mismo sistema, sin problemas de conflicto de puerto.

03

Conjunto de protocolos TCP/IP

Como administrador de red, usted requerirá interactuar directamente con los protocolos TCP/IP y servicios relacionados. Para esto necesitará una buena comprensión de TCP/IP, de los protocolos asociados y de sus aplicaciones. Hoy en día, el conjunto de protocolos TCP/IP lo usan ampliamente muchas corporaciones, instituciones y otras organizaciones, para abordar sus necesidades de trabajo en red. Éste es considerado la familia de protocolo preferida de los fabricantes de redes, los vendedores de sistemas operativos y los usuarios, por igual. En efecto, la red más grande del mundo, Internet, usa exclusivamente TCP/IP.

Conjunto de protocolos TCP/IP

Si usted ha tenido acceso a un sitio Web, transferido archivos desde un servidor FTP o enviado un correo electrónico a través de Internet, ha usado, de manera indirecta, los protocolos TCP/IP. Fundamentalmente, TCP/IP suministra una forma estándar para entregar datos desde un sistema a otro, sin importar las diferencias del sistema operativo y del hardware de red. TCP/IP es un acrónimo que representa dos protocolos separados: el protocolo de control de transmisión (TCP) y el protocolo Internet (IP). Sin embargo, TCP/IP generalmente se refiere a estos protocolos, más un conjunto de protocolos relacionados. Por ejemplo, el protocolo de transferencia de archivos (FTP) usa el TCP/IP y provee una facilidad básica de transferencia de archivos a través de los dispositivos que soportan TCP/IP. Si el dispositivo soporta TCP/IP, se supone por lo general que también soporta FTP, un *host* de protocolos y otros servicios.

Hoy en día, el conjunto TCP/IP es soportado en todo sistema operativo de computadora disponible. Como tal, es considerado el protocolo de red más popular, y muchos de los mismos servicios TCP/IP están disponibles en diferentes versiones de UNIX. Ésta es una buena noticia, porque muchas de las funciones centrales del TCP/IP y las aplicaciones son las mismas a través de diferentes versiones de UNIX. Las operaciones del TCP/IP son independientes del sistema operativo o de la plataforma de cómputo. El protocolo oculta los detalles del sistema operativo subyacente y proporciona un marco de trabajo común para establecer conectividad entre los sistemas.

Muchos dispositivos de trabajo en red como los enrutadores, conmutadores y dispositivos especializados, implementan los protocolos TCP/IP para la administración y el control locales. En general, un pequeño conjunto central de servicios está disponible en estos dispositivos. La tabla 3.1 muestra la lista de servicios disponibles en algunos de los dispositivos de trabajo en red más populares. Como se puede observar, SNMP, Telnet y FTP son los servicios más ampliamente implementados.

El conjunto TCP/IP está construido sobre estándares de la industria, y es documentado completamente en las Solicitudes de Comentarios (RFC). Esto significa que el protocolo se mantendrá abierto y estandarizado, y ningún vendedor individual podrá poseer los protocolos o desarrollar extensiones propietarias. Por ello, muchos vendedores de hardware de trabajo en red suministran soporte para TCP/IP en sus productos.

Tabla 3.1 Soporte de protocolo TCP/IP para administración.

Tipo de dispositivo	Protocolos TCP/IP
Conmutador CoreBuilder 3Com	HTTP, FTP, Telnet y SNMP
Sonda RMON 3Com	TFTP y SNMP
Enrutador Cisco 7000	HTTP, FTP, Telnet y SNMP
Modificador de canal de fibra Brocade	HTTP, Telnet y SNMP
Impresora láser Lexmark	FTP y SNMP

TCP/IP es independiente del protocolo de enlace de datos y puede usarse con muchas tecnologías de trabajo en red diferentes, incluidas FDDI, Ethernet, ATM, TokenRing, Frame Relay y SMDS. TCP/IP hace posible construir una verdadera red heterogénea que consta de productos y tecnologías de red de muchos vendedores y fuentes diferentes. En efecto, Internet, considerada la red más grande del mundo, consta de dispositivos de muchos vendedores de trabajo en red que operan juntos. Esto no quiere decir que Internet no tenga participación en los temas y problemas del trabajo en red, pero, por lo general, muchos estarán de acuerdo en que la interoperabilidad entre los vendedores de equipos no es el factor principal para el conjunto central establecido de los protocolos TCP/IP.

Desde el punto de vista de la administración, muchas de las herramientas utilizadas en la administración de sistemas TCP/IP son consistentes con la mayoría de las versiones del sistema operativo UNIX. Estas herramientas son descritas en el capítulo 6, "Vista general de MIB-II". Sin embargo, uno de los aspectos que puede ser un problema es que cada uno de los vendedores de sistemas operativos UNIX puede implementar de forma diferente los detalles que no son del protocolo. Por ejemplo, ningún estándar RFC cubre la forma como es almacenada la información de la dirección IP, y esto no debería ser así. El presente conjunto TCP/IP proporciona un mecanismo que asigna dinámicamente direcciones IP a los dispositivos, y también obliga que sean asignadas sólo a cada dispositivo adjunto a la red. Sin embargo, las direcciones IP son almacenadas en un sistema local y no son problema del protocolo, pero sí son un asunto de configuración del

sistema o de la administración de red, lo cual es tradicionalmente resuelto en el nivel de sistema operativo. Cada uno de los vendedores del sistema operativo suministra sus propias soluciones en relación con la manera como deben ser almacenada la información de la dirección IP u otros parámetros del sistema operativo.

Modelo OSI

Los protocolos de trabajo en red, incluido TCP/IP, pueden ser trasladados a un modelo general de red. Este modelo define la relación y los servicios que cada protocolo suministrará a otros protocolos, servicios y aplicaciones asociados. El modelo estándar de redes más común está basado en la interconexión de sistemas abiertos (OSI). El modelo OSI de siete capas se representa mediante una serie de capas colocadas unas sobre otras, las cuales, cuando son vistas colectivamente, representan la operación de una red completa. Cada capa representa una vista única de los elementos que componen la red. Las capas del modelo OSI son las siguientes:

- aplicación
- presentación
- sesión
- transporte
- red
- enlace de datos
- física

Capa de aplicación

Suministra servicios a los usuarios, que incluyen transferencia de archivos, correo electrónico, acceso remoto a *host* y muchos más.

Capa de presentación

Se utiliza para suministrar una interfaz común para aplicaciones en las capas inferiores, e implementar servicios comunes que pueden incluir encriptación, reformateado y compresión.

Capa de sesión

Suministra el mecanismo para establecer, mantener y terminar sesiones entre aplicaciones cooperativas.

Capa de transporte

Asegura la confiabilidad, la transferencia transparente de datos, el control de flujo y la recuperación y detección de errores de datos entre dos puntos finales.

Capa de red

Suministra transparencia en el protocolo de la capa superior, debido a que pueden utilizarse diferentes metodologías de comunicación en red. Esta capa es responsable de establecer, mantener y terminar conexiones para redes diferentes.

Capa de enlace de datos

Suministra el servicio de transferencia de datos en el enlace físico usando marcos; maneja la detección de errores, el control de flujo y servicios relacionados. Algunos tipos comunes de marcos incluyen Ethernet, FDDI y TokenRing.

Capa física

Aborda los requerimientos del mecanismo de conectividad (como cables y conectores) y proporciona la transmisión de un pequeño flujo que involucra las características de control de voltaje para producir las señales apropiadas. Los ejemplos incluyen cableado para Ethernet, Fast Ethernet, FDDI y TokenRing.

Arquitectura del protocolo TCP/IP

Colocar o trasladar al conjunto TCP/IP en el modelo OSI ayuda a demostrar la operación del protocolo TCP/IP y su relación con otros protocolos. La figura 3.1 muestra dónde encaja TCP/IP en el modelo OSI.

Como se muestra en la figura 3.1, el modelo TCP/IP consta de cuatro capas. Cada capa es transformada a una o más de las capas OSI, las cuales incluyen:

- proceso
- *host a host*
- Internet
- acceso a la red

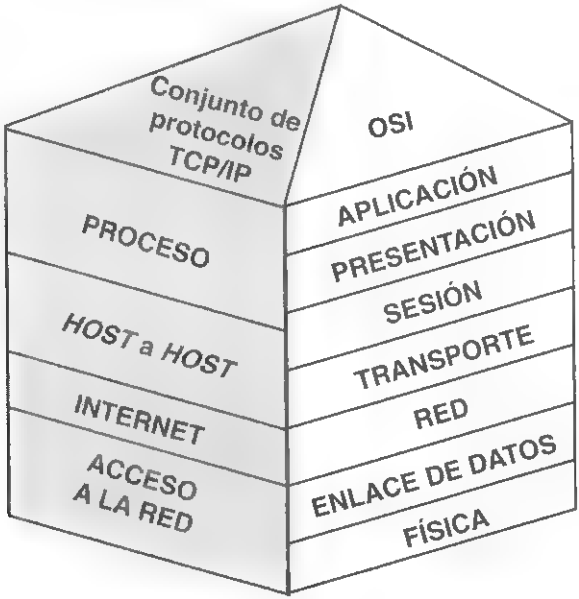


Figura 3.1 TCP/IP sobre el modelo OSI de trabajo en red.

Todas las capas componentes, excepto la de acceso a la red, están basadas en software y consisten en módulos programados que suministran la funcionalidad requerida. Comúnmente, estos componentes están incorporados en los sistemas operativos para proporcionar acceso generalizado.

Capa de proceso

Esta capa suministra aplicaciones e interfaces de usuario con la capa *host* a *host*. En esta capa también se encuentran protocolos y servicios adicionales. La capa de proceso se traslada a las capas de presentación y aplicación, las cuales están definidas dentro del modelo OSI. Las aplicaciones en esta capa incluyen Telnet, FTP, SMTP y muchas otras.

Capa *host* a *host*

Esta capa es responsable de asegurar la confiabilidad de los da tos y de que cada servicio de alto nivel obtenga la información correcta de la entidad emisora. El protocolo soportado en esta capa es TCP. La capa se traslada a la capa OSI de transporte. El término *segmento* se usa para describir la información en la capa *host* a *host*.

Capa Internet

Esta capa proporciona un flujo no confiable de información desde una red hasta otra. Desde un punto de vista OSI, esta capa se define como la capa de red. La capa Internet (o de red) es responsable del enrutado entre diferentes redes IP. El protocolo soportado en esta capa es IP. El término *paquete* se usa para describir la información procesada en esta capa.

Capa de acceso a la red

La capa de acceso a la red incluye conexiones físicas a la red, lo cual, tradicionalmente requiere una interfaz de hardware desde la red hasta un bus interno de la computadora. La capa incluye la capa física y la capa de enlace de datos del modelo OSI. El componente de acceso a la red define la arquitectura y la topología de la red. Algunos ejemplos incluyen Ethernet, FDDI y TokenRing. El término que se usa para describir la información procesada en esta capa es *marco* (*frame*). También se nesecita un pequeño programa controlador, el cual es suministrado por los fabricantes de la interfaz, para relacionar el hardware con el sistema operativo.

Los protocolos TCP/IP y los protocolos asociados pueden también ser colocados en el modelo OSI de siete capas, como se muestra en la figura 3.2. En ella podemos ver los protocolos y servicios relacionados con el modelo. En las siguientes páginas se describen muchos servicios adicionales suministrados por el conjunto de protocolos TCP/IP.

Servicios de la capa de proceso

Los servicios TCP/IP en la capa de proceso incluyen herramientas de usuario final, protocolos adicionales y servicios del sistema. Incluido en las diferentes plataformas UNIX, TCP/IP suministra un mecanismo común para compartir archivos, enviar/recibir correo electrónico, tener acceso a sistemas de forma remota, transferir archivos entre sistemas y realizar otras tareas del trabajo en red. Aunque el protocolo TCP/IP y el conjunto de aplicación es bastante grande, muchos vendedores del sistema UNIX proporcionan un subcojunto más pequeño de estos servicios. La tabla 3.2 enumera muchos de los servicios centrales TCP/IP generalmente disponibles en los sistemas UNIX.

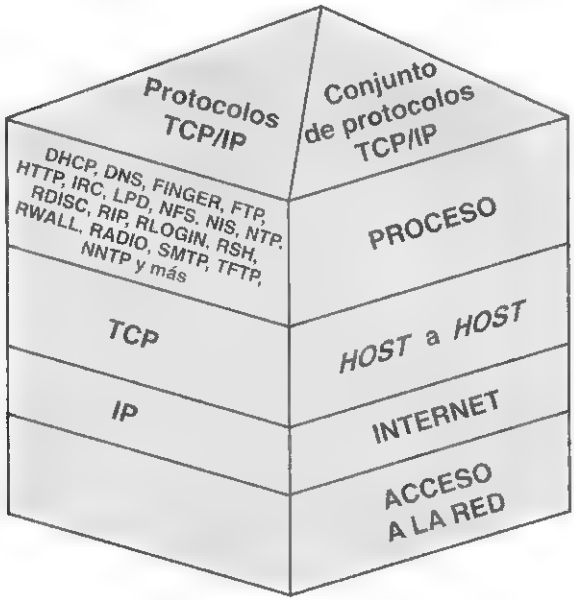


Figura 3.2 Conjunto de protocolos TCP/IP.

Herramientas de usuario final

Las herramientas de usuario final, las cuales son comunes a muchas implementaciones del sistema UNIX de TCP/IP, son aplicaciones que están generalmente disponibles para usuarios normales del sistema. Como resultado, estas herramientas no requieren privilegios de root para su operación. Por ejemplo, los usuarios generales, sin ninguna consideración especial desde el punto de vista de la administración, pueden invocar los comandos Telnet y FTP. Algunos servicios en el conjunto TCP/IP se refieren tanto a las aplicaciones de usuario final como a los protocolos. Telnet es un buen ejemplo de esto porque representa tanto una herramienta de usuario como un protocolo de comunicación. En la práctica, sin embargo, éste no es un gran problema, porque las aplicaciones del usuario final en UNIX están en minúsculas (como `telnet`) y los protocolos son escritos generalmente en mayúsculas.

Es interesante notar que ciertas organizaciones inhabilitan algunos servicios TCP/IP como una forma de intensificar la seguridad. Una organización en particular no querría que sus usuarios tuvieran la habilidad de enviar o recibir correo electrónico

en los sistemas de desarrollo centrales y remueven los servidores SMTP de esos sistemas. Otra forma como las organizaciones comúnmente inhabilitan los servicios, es por medio del bloqueo de acceso a los puertos del sistema usando un dispositivo de barrera de seguridad (*firewall*) o enrutador.

Protocolos adicionales

El conjunto TCP/IP incluye protocolos de alto nivel que existen sobre la capa de red y suministran los detalles necesarios para asegurar que las aplicaciones puedan comunicarse. Por ejemplo, el protocolo de transferencia de archivos (FTP) define cómo son transmitidos a través de la red los archivos e información asociada. El protocolo maneja todos los detalles relacionados con la autorización del usuario, nombres y representación de datos entre sistemas heterogéneos.

Servicios del sistema

Los servicios del sistema TCP/IP incluyen esas facilidades que son suministradas a todos los usuarios del sistema, pero que sólo pueden ser controladas por el administrador del sistema. Los servicios del sistema incluyen procesos específicos del sistema y archivos especiales de configuración usados por esos procesos. Los servicios del sistema de red son, a menudo, arrancados automáticamente cuando el sistema arranca o cuando el sistema es reiniciado, pero algunos arrancan en respuesta a solicitudes.

Tabla 3.2 Servicios y protocolos centrales TCP/IP.

Servicio	Descripción
ARP/RARP	Protocolo de resolución de direcciones Protocolo invertido de resolución de direcciones
DHCP	Protocolo de configuración dinámica del *host*
DNS	Servicio de nombres de dominio
FINGER	Búsqueda de usuario remoto/local
FTP	Protocolo de transferencia de archivos

Tabla 3.2 Protocolos y servicios centrales TCP/IP. (*continuación*)

Servicio	Descripción
HTTP	Protocolo de transferencia de hipertexto
ICMP	Protocolo de mensajes de control en Internet
LPD	Demonio de impresora de línea
NFS	Sistema de archivos de red
NIS	Servicio de información de red
NTP	Protocolo de tiempo de red
RDISC	Protocolo de descubrimiento de enrutador
REXEC	Servicio de ejecución remota
RIP	Protocolo de información de enrutado
RLOGIN	Servicio de inicio de sesión remota
RPC	Llamada a procedimiento remoto
RSH	Servicio de shell remoto
RWHO	Seguimiento de usuarios remotos
RWALL	Emisión remota de mensajes
RADIO	Radio transmisor/receptor
SMTP	Protocolo simple de transferencia de correo
TALK	Hablar con usuarios locales/remotos
Telnet	Acceso a sistemas remotos
TFTP	Protocolo trivial de transferencia de archivos
WHOIS	Servicio de consulta remota

Los protocolos y servicios centrales TCP/IP enumerados en la tabla 3.2 son definidos además como sigue:

ARP: el protocolo de resolución de direcciones suministra la transferencia entre los protocolos de bajo nivel de enlace de datos (como Ethernet y TokenRing) y los protocolos de alto nivel como IP. El ARP traslada direcciones de enlace de datos (es decir, interfaz de hardware) a direcciones IP. El protocolo invertido de resolución de dirección (RARP) se usa para ir en el otro sentido; traslada direcciones IP a direcciones de protocolo de enlace de datos. ARP y RARP se describirán completamente más adelante en esta sección.

DHCP: el protocolo de configuración dinámica de *host* proporciona información de inicio (arranque) a los sistemas cliente. El DHCP soporta información de direcciones IP, información de configuración del sistema operativo y otra información relacionada. Desde el punto de vista de una dirección de red, el DHCP es una excelente forma para administrar direcciones IP a través de la empresa. Por ejemplo, los clientes pueden obtener, de una forma dinámica, información de la IP durante el arranque, evitando así el problema de tener que configurar cada máquina.

DNS: el servicio de nombres de dominio traslada los nombres de *host* y las direcciones IP. El lado cliente suministra la habilidad de resolver nombres y direcciones por medio de solicitudes a uno o más de los servidores DNS, solamente. El componente al lado del servidor, `named`, escucha las solicitudes y busca entradas en una base de datos local o contacta a otro servidor de resolución de nombres.

FINGER: los servicios *finger* permiten la consulta de información del usuario, tanto en un sistema local como en uno remoto. El servicio finger no es un protocolo, sólo es un programa de usuario final que utiliza TCP para la comunicación con el servidor `in.fingerd`.

FTP: el protocolo de transferencia de archivos transfiere archivos entre sistemas. El FTP suministra autorización básica del usuario, que incluye el uso del nombre de sesión y de la clave (*password*) en el sistema remoto. La interfaz FTP es básica, pero proporciona una forma simple para transferir archivos múltiples o uno solo. El servidor FTP es conocido como `in.ftpd`. FTP soporta la transmisión tanto de los archivos de datos binarios como de los archivos de datos ASCII.

HTTP: el protocolo de transferencia de hipertexto transmite páginas Web y documentos de un servidor de Web a un explorador.

ICMP: el protocolo de mensajes de control de Internet es una facilidad de diagnóstico de red que utiliza el protocolo IP. La herramienta `ping` utiliza el protocolo ICMP eco solicitud/respuesta para determinar la conectividad del nodo.

LPD: el dominio de impresora de línea suministra una facilidad de impresión, tanto para la red como para impresoras conectadas directamente.

NFS: la facilidad del sistema de archivos de red permite compartir archivos entre sistemas en una red local.

NIS: el servicio de información de red es una facilidad de consulta de directorio que suministra al cliente acceso a bases de datos del servidor. Los tipos de información comúnmente usados en NIS incluyen inicio de sesión, *host*, archivos compartidos y otra información de configuración del sistema.

NTP: el protocolo de tiempo de red proporciona una excelente forma de asegurar que la información de tiempo y fecha está sincronizada entre todos los sistemas de red UNIX.

RDISC: el protocolo de descubrimiento de enrutador de red ICMP encuentra enrutadores y construye una tabla de rutas a redes adjuntas.

REXEC: el servicio de ejecución remota proporciona la ejecución de comandos UNIX en sistemas remotos. El REXEC utiliza un procedimiento especializado de autenticación que incluye la lectura de inicio de sesión y de la clave, y compara esta información con el sistema remoto. Si la información de inicio de sesión coincide, se ejecuta el comando UNIX. La familia de comandos remotos incluye **rsh**, **rwho**, **rlogin** y otros.

RIP: el protocolo de información de enrutado propaga información de enrutado entre los dispositivos del sistema de red, como los enrutadores. Los sistemas UNIX soportan también al RIP. En algunos sistemas UNIX, si dos o más interfaces de red están instaladas, el sistema desempeñará automáticamente funciones de enrutado. La función de enrutado es incorporada en el proceso del sistema `in.routed`, que se pone en marcha cuando el sistema es iniciado.

RLOGIN: el servicio remoto de inicio de sesión da acceso a un sistema remoto UNIX. Suministra los mismos servicios básicos que el programa Telnet.

RPC: la llamada a procedimientos remotos es un mecanismo y un protocolo que permite la ejecución de procedimientos a través de la red, de manera neutral.

RSH: el servicio de shell remoto suministra un shell al sistema remoto.

RWHO: el RWHO suministra una lista de usuarios conectados a un sistema remoto. Este comando es similar al comando **who** de UNIX.

RWALL: el RWALL suministra una forma para escribir a los usuarios en un sistema remoto. Este comando es similar al comando **wall** de UNIX.

RADIO: ésta es la facilidad de difusión de radio.

SMTP: el protocolo simple de transferencia de correo suministra el mecanismo de entrega de correo que usan muchos paquetes de correo electrónico y es el protocolo estándar de correo de Internet. El programa de sistema `sendmail` implementa SMTP y es responsable de la propagación de correo entre sistemas.

TALK: Talk es una facilidad de comunicación bilateral que se puede usar para hablar con otros usuarios del sistema, tanto en un sistema local como en un sistema remoto. Talk no es un protocolo, es sólo una utilidad del sistema de usuario final que utiliza el protocolo UDP.

Telnet: Telnet es el nombre de un protocolo y una utilidad del sistema de usuario final. La utilidad Telnet suministra una interfaz de usuario a un sistema remoto UNIX. Los usuarios pueden entrar en otros sistemas sobre la red y ejecutar comandos como si ellos fueran locales para ese sistema. Su terminal está conectado a través del protocolo Telnet al sistema remoto, usando el proceso del servidor `in.telnetd`. El protocolo Telnet define una interfaz virtual de red que controla el flujo y la interpretación de un flujo de caracteres entre sistemas.

TFTP: el protocolo trivial de transferencia de archivos suministra una facilidad de transferencia de archivos más sencilla que el FTP. TFTP es considerado una versión ligera del FTP porque no soporta un mecanismo de autorización robusto o comando set. El TFTP se utiliza principalmente para descargar información de configuración o datos del sistema.

WHOIS: el WHOIS es una utilidad de consulta de páginas blancas. El servicio WHOIS buscará usuarios individuales y otra información desde servidores estándar de Internet.

Servicios adicionales

Muchos servicios y aplicaciones de dominio públicoTCP/IP están también disponibles a través de la Internet. Algunos de los recursos disponibles son mejoras del conjunto central de servicios existente, mientras que otras aplicaciones suministran nuevos sevicios y características. La tabla 3.3 enumera algunas de estas aplicaciones TCP/IP.

Tabla 3.3 Servicios adicionales TCP/IP.

Servicio	Descripción
ARCHIE	Facilidad de búsqueda FTP
GOPHER	Sistema de recuperación de documentos
IRC	Servicio de conversación en tiempo real en Internet
NNTP	Protocolo de transferencia de noticias en red
WAIS	Servidor de información de área amplia

ARCHIE: archie es una base de datos de sitios FTP anónimos y sus contenidos. Archie sigue la pista del contenido total de un gran número de sitios FTP anónimos, y permite buscar archivos en esos sitios usando varios tipos de búsquedas de nombres de archivos.

GOPHER: GOPHER es un sistema de recuperación de documentos que está disponible a través de la interfaz manejada por menús (para dispositivos basados en caracteres) y la World Wide Web (WWW).

IRC: la conversación en tiempo real en Internet es una forma de enviar mensajes públicos o privados de texto en tiempo real a uno o más suscriptores.

NNTP: el protocolo de transferencia de noticias en red proporciona la habilidad para transferir archivos de noticias (también conocido como Usenet) entre un cliente y un servidor.

WAIS: WAIS es otra facilidad que ayuda a buscar material y archivos indexados.

Capa *host* a *host*

La capa *host* a *host*, o capa de red OSI, es responsable de suministrar un mecanismo robusto de entrega de datos entre diferentes entidades de red. El estándar que proporciona este servicio es el protocolo de control de transmisión (TCP). Dentro de una red, los datos pueden perderse o destruirse cuando ocurren errores de transmisión o fallas en el hardware de red. Los datos también

pueden ser entregados en desorden y con demoras significativas antes de que alcancen su destino final. El TCP fue diseñado y desarrollado para solucionar este tipo de problemas relacionados con la red. El TCP es responsable de asegurar que los datos lleguen en el orden correcto y libres de errores. Éste cumple estas tareas suministrando los servicios descritos enseguida.

Conexiones virtuales

El TCP provee una interfaz de conexión virtual a la red que es análoga a la forma en que las llamadas de teléfono se establecen en la red telefónica. Conceptualmente, un usuario llama a otra máquina para solicitar transferencia de datos. Después de que todos los detalles de la configuración de la conexión se completan, la transmisión de datos puede ocurrir entre las aplicaciones. Desde la perspectiva de una aplicación, la conexión TCP se ve y comporta como si se hubiera establecido un enlace dedicado de hardware. Sin embargo, esto es sólo una ilusión suministrada por la interfaz de flujos TCP.

Datos en secuencia

Para asegurar la transferencia confiable, el TCP sigue la pista de los datos que transmite asignando un número de secuencia a cada segmento. El número de secuencia únicamente identifica cada segmento de datos dentro de una conexión y proporciona un reconocimiento positivo a la entidad emisora. Ningún reconocimiento indica que el mensaje debería ser retransmitido. El número de secuencia se usa también para reordenar algunos segmentos que puedan haber llegado en desorden. ¿Cómo pueden llegar en desorden los segmentos? Considérese, por ejemplo, la red en la figura 3.3.

Debido a que existe más de una ruta de red al `Nodo C`, es posible que algunos segmentos TCP puedan viajar a través del enrutador `R2`, en lugar del enrutador `R1`. Por ejemplo, la ruta entre el `Nodo C` y `R1` podría estar temporalmente cargada de forma excesiva, y los segmentos pueden ser enrutados a través de un camino alterno. Como resultado, los segmentos que utilizan la ruta `R2` pueden llegar a su destino antes que los segmentos que usan la ruta `R1`.

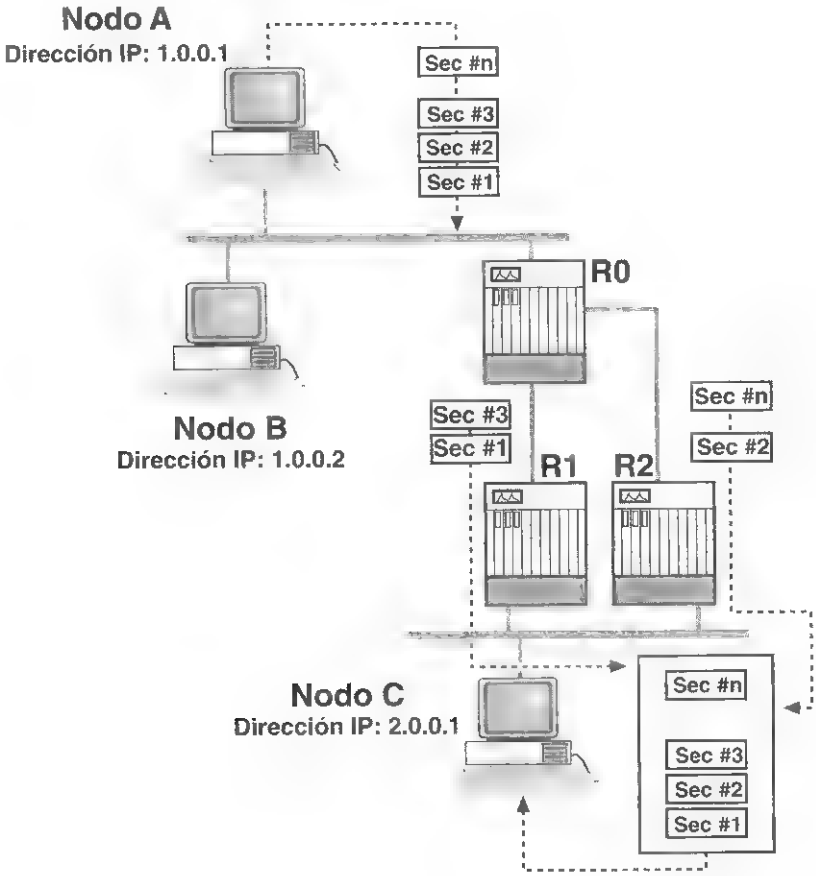


Figura 3.3 Números de segmento TCP en acción.

Interfaz de abstracción de flujo

Desde el punto de vista de la capa de aplicación, el TCP suministra una interfaz con buffer orientada al byte entre dos aplicaciones o procesos. Los datos transmitidos desde la entidad fuente son exactamente la misma información que el destinatario recibe. Por ejemplo, si la entidad emisora transmite el mensaje `Hello World`, el destinatario recibirá `Hello World`. Ésta es una característica muy útil y conveniente para el desarrollo de aplicaciones y servicios de trabajo en red. Además, el flujo TCP está provisto de buffers, lo cual significa que las aplicaciones tienen más flexibilidad cuando se procesan los datos de la red.

Puertos, zócalos y conexiones

Los puertos TCP son direcciones que especifican un recurso de red, y se emplean para identificar inequívocamente una aplicación individual o un servicio del sistema. Hay muy pocas direcciones de puertos conocidas en uso hoy en día, y muchas de ellas se pueden encontrar en el archivo `/etc/services` en sistemas UNIX. *Véase* el apéndice B para información adicional respecto a este archivo. La tabla 3.4 contiene una lista parcial de algunos de los puertos TCP más comúnmente usados.

Tabla 3.4 Puertos TCP comunes.

Puerto	Aplicación/servicio
20	Datos FTP
21	FTP
23	Telnet
25	SMTP
53	DNS
119	NNTP
161	SNMP
80	HTTP

Para una mejor comprensión de la función de estos puertos, considérense los servicios del proceso UNIX `inetd`. Este proceso es llamado el superservidor de Internet, porque es responsable de conectar las solicitudes de servicio desde la red hasta el verdadero programa servidor. `Inetd` sabe cuál proceso invocar, porque puede determinar las relaciones entre puertos y servicios. Por medio del procesamiento del archivo `/etc/services` y el archivo de configuración `/etc/inetd.conf`, `inetd` puede hacer la solicitud de red al servicio apropiado que se necesite. El apéndice B contiene información adicional respecto al archivo `inetd.conf`. La figura 3.4 muestra la operación de `inetd` cuando un usuario remoto solicita una sesión Telnet.

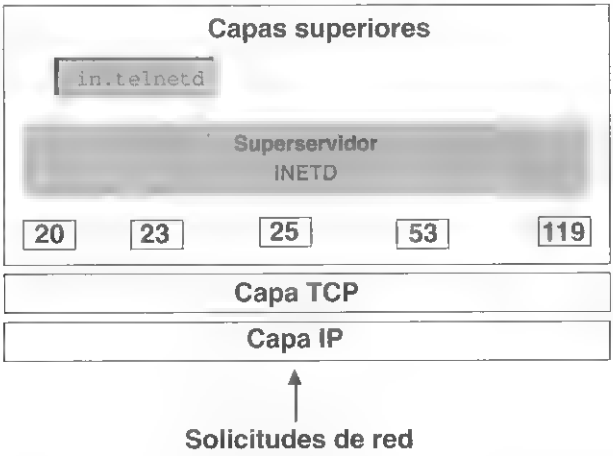


Figura 3.4 Operación de INETD con solicitud Telnet.

Es importante entender que el TCP usa un modelo orientado a la conexión, por la cual una entidad de red puede llamar a otra para establecer una sesión semidúplex o dúplex completa. En el modo duplex completo, se establecen dos canales de red independientes entre los sistemas, y los datos pueden fluir entre los sistemas, sin una interacción aparente. En el modo semidúplex, solamente se establece un único canal. Una entidad de red puede establecer primero una sesión dúplex completa y luego apagar un canal, si es necesario. Por otra parte, un servicio puede establecer inicialmente una única sesión semidúplex, para fines de control, y luego arrancar otro canal para llevar a cabo alguna acción o tarea específica. El comportamiento de esta aplicación puede parecer un poco extraño, pero el servicio FTP, por ejemplo, opera de esta manera.

Cuando comienza una sesión FTP, ésta establece primero una única sesión al sistema destino. Esta sesión se utiliza para la autenticación del usuario y la interfaz de comando. Cuando el usuario especifica la transferencia de un archivo o ejecuta un comando remoto, se establece otra sesión para atender la solicitud. Después de que la transferencia se completa, la sesión nueva se cierra. Este proceso se repite para cada transacción por separado.

Los zócalos son puertos que el sistema asigna en nombre del usuario cuando se ejecutan las aplicaciones de red u otros servicios.

Debido a que el sistema operativo genera un número único de zócalo, dos aplicaciones que se ejecutan simultáneamente no tendrán el mismo número de zócalo. En algunos sistemas UNIX, la asignación de zócalos empieza por encima del 1024.

En el contexto de una conexión, el TCP usa cuatro piezas de información para identificar inequívocamente una sesión de otra: dirección fuente IP, puerto fuente, dirección de destino IP y puerto de destino. Esto es importante de recordar, porque pueden establecerse muchas sesiones para la misma aplicación o servicio, aún desde el mismo *host*. Por ejemplo, dos usuarios diferentes pueden hacer `telnet` al mismo *host* de destino, sin conflictos entre los puertos. Esto se consigue por el hecho de que el TCP usa todos los cuatro elementos de dirección para distinguir una sesión única. La figura 3.5 muestra la relación de los elementos TCP en diferentes sesiones.

	Entidad fuente IP: 10.0.2.1	Entidad destino IP: 10.0.2.1
Conexión TCP #1	Dirección fuente: 10.0.2.1 Puerto TCP: 1001	Dirección destino: 10.0.2.2 Puerto TCP: 23
Conexión TCP #2	Dirección fuente: 10.0.2.1 Puerto TCP: 1002	Dirección destino: 10.0.2.2 Puerto TCP: 23

Figura 3.5 Dos sesiones TCP desde la misma fuente/destino.

Reconocimiento positivo

Una forma en que el TCP proporciona confiabilidad es asegurando que todo mensaje transmitido sea confirmado por la entidad receptora. La confirmación de los mensajes TCP es conocida como reconocimiento positivo, y se usa para asegurarse de que la entidad receptora ha obtenido todos los segmentos que se han enviado. Cuando se envía un mensaje TCP, la entidad emisora arranca un temporizador. Si no se recibe ningún reconocimiento antes de que el tiempo haya expirado, TCP supone que el mensaje se perdió o dañó en el camino, impidiendo su entrega. Como resultado, TCP envía otro mensaje para remplazar el primero y arranca otra vez el proceso del temporizador. Este proceso continúa hasta que todos los segmentos han sido reconocidos o hasta que se alcanza un umbral interno de error. Si el emisor no recibe ningún reconocimiento de segmentos pendientes después de que

el conteo interno de error se ha alcanzado, la conexión se terminará.

Estableciendo y cerrando una conexión TCP

Como se analizó previamente, TCP utiliza conexiones que proporcionan una facilidad de transferencia de datos robusta y confiable. El procedimiento para establecer o cerrar una conexión no es un proceso mágico. En cambio, cada entidad TCP sigue el mismo conjunto de reglas cuando se crea o se termina una sesión. Para establecer una conexión, TCP utiliza un protocolo de acuerdo de conexión de tres vías, que se expone a grandes rasgos en la figura 3.6.

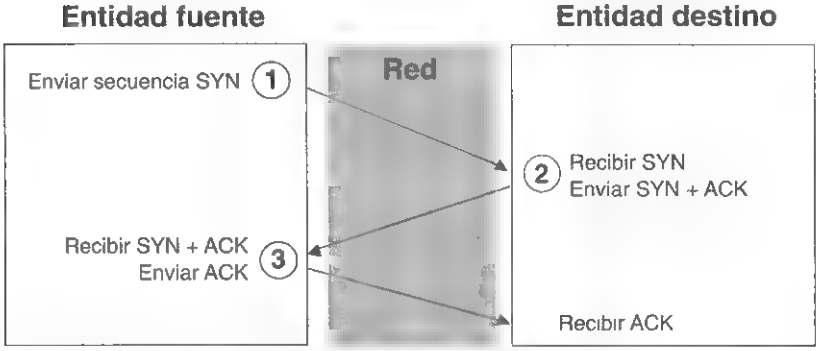


Figura 3.6 Apertura de una conexión TCP mediante el uso de un acuerdo de conexión de tres vías.

Primero, la fuente transmite un segmento de mensaje SYN. La SYN (pronunciado "sin") o sincronización, es una solicitud para arrancar una sesión TCP y tiene el bit SYN establecido en el campo código. A continuación, el destinatario o receptor responde con un segmento ACK que tiene tanto el bit SYN como los bits ACK establecidos en el campo código, indicando que la solicitud ha sido aceptada y que continúa el protocolo de acuerdo de conexión. Finalmente, la fuente envía un segmento ACK, el cual informa al destino que ambas entidades están de acuerdo en que se ha establecido una conexión y que los segmentos pueden ser ahora transmitidos y recibidos.

Para cerrar una sesión establecida, TCP utiliza un acuerdo modificado de tres vías, como se muestra en la figura 3.7. Primero, la

fuente transmite un segmento FIN o final (el bit FIN es establecido en el campo código) como resultado de la aplicación, queriendo cerrar su lado de la conexión. Recordando que TCP ve estas conexiones como dúplex completo, entonces, cualquier parte puede terminar su lado de la conexión. Una vez la aplicación en el destino cierra la conexión, TCP emite un segmento FIN a la fuente. A continuación, la fuente recibe la secuencia FIN y envía un reconocimiento.

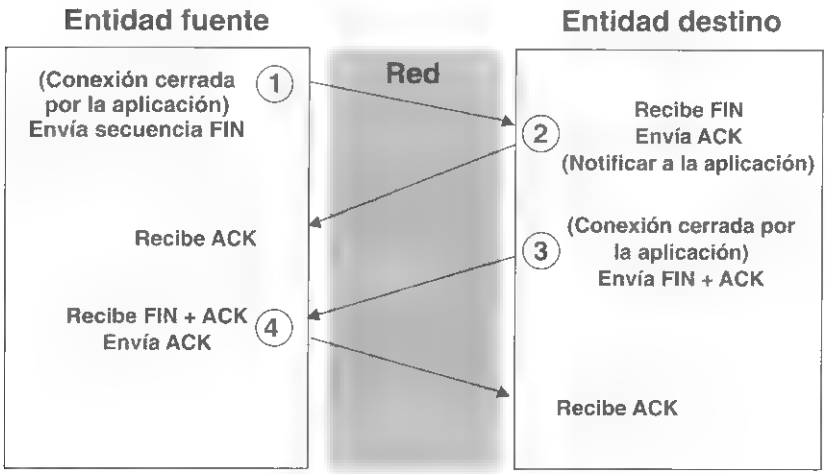


Figura 3.7 Cierre de una conexión TCP.

Nótese que esto toma tres segmentos para crear una conexión TCP y cuatro segmentos adicionales para cerrarla. Se requiere un total de siete mensajes para operar una conexión TCP, sin incluir ningún segmento de transferencia de datos.

Máquina de estados

La operación del TCP se describe mejor usando un modelo de máquina de estados, el cual controla la operación básica del protocolo. La figura 3.8 muestra una ilustración representativa de la máquina de estados TCP, donde cada conexión TCP debe seguir una serie de fases definidas. El movimiento de un estado a otro es el resultado de un evento o una transición. La etiqueta en cada transición muestra qué recibe TCP para causar el cambio entre los estados. Por ejemplo, analizamos que el TCP debe abrir una conexión antes de que los datos puedan ser transferidos.

Normalmente, cada lado del TCP de la conexión arranca en el estado CERRADO. Cuando se desea una conexión, se hace una transición desde el CERRADO hasta el estado ENVIAR SYN. En este punto, el lado del cliente envía un paquete SYN. Si el paquete SYN es aceptado, el lado remoto emite un ACK, el cual causa una transición desde ENVIAR SYN hasta el estado SYN RECIBIDO. Una vez se ha recibido el último ACK, se alcanza el estado ESTABLECIDO y la transferencia de datos puede comenzar. Cuando se hace una conexión TCP, se mantiene en el estado ESTABLECIDO hasta que cualquiera de los dos lados quiera terminar la conexión, o hasta que ocurra algún error.

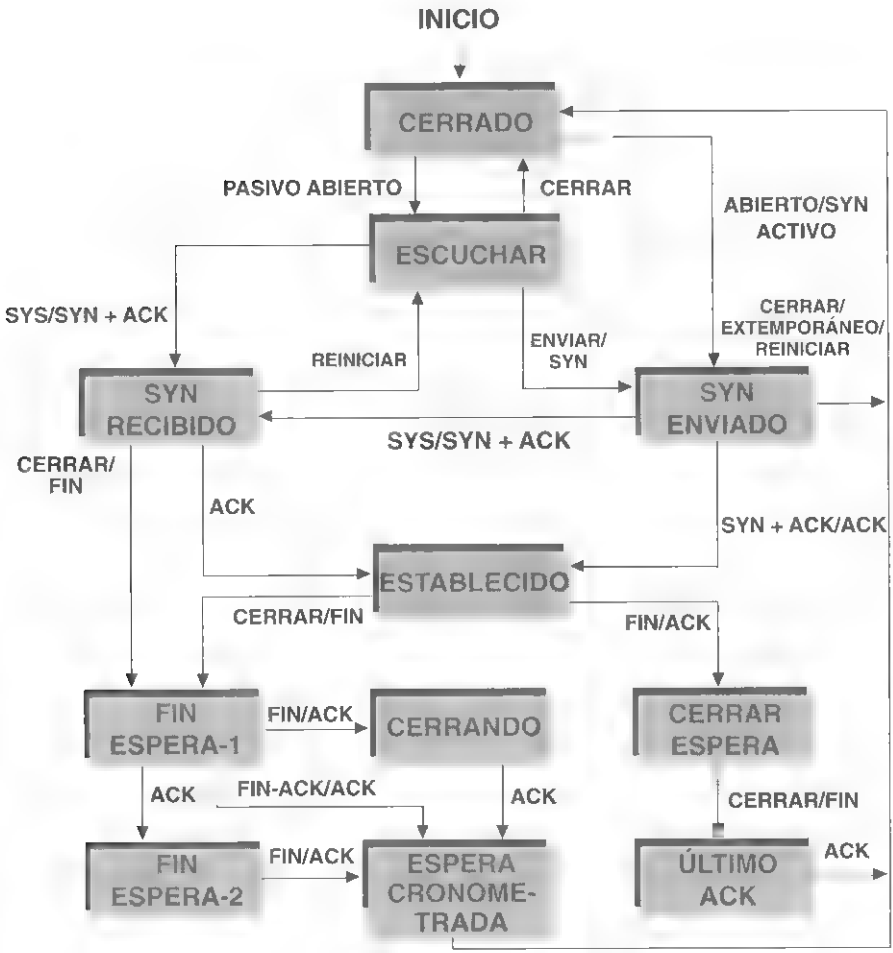


Figura 3.8 Máquina de estado TCP.

Cuando se termina una conexión TCP (ya sea por la fuente o por el destino), la conexión se mueve hacia el estado CERRAR ESPERA o FIN ESPERA-1. Si la fuente envía un segmento FIN, el TCP realiza una transición al estado CERRAR ESPERA, el cual finalmente termina la conexión. Cuando el destino quiere cerrar la conexión, se hace un cambio al estado FIN ESPERA 1. El TCP tiene un mecanismo elaborado para asegurarse de que los segmentos de conexiones previas no interfieran con las existentes. El TCP mantiene un temporizador, conocido como tiempo de vida máximo del segmento (MSL), el cual contiene el tiempo máximo que un segmento caducado puede mantenerse vivo dentro de la red. Como resultado, el TCP se mueve al estado ESPERA CRONOMETRADA después de cerrar la conexión. Éste mantiene este estado por dos MSL. Después de esto, si algunos segmentos llegan para la conexión, son rechazados. ¿Por qué nos deben importar estos detalles? Bueno, algunas veces es provechoso en la localización de problemas de conectividad. El comando **netstat**, por ejemplo, suministra información del estado del puerto TCP para ayudar con este tipo de actividad.

Formato de secuencia TCP

El TCP define un formato de secuencia, el cual incluye toda la información necesaria para asegurarse de que los segmentos lleguen al destino correcto, e información adicional de control. La figura 3.9 muestra el formato del segmento TCP.

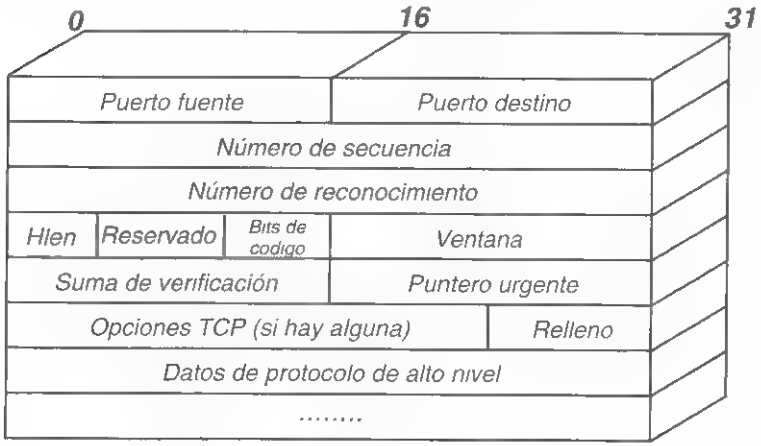


Figura 3.9 Formato del segmento TCP.

Los campos del segmento TCP incluyen:

Puerto fuente:	es el protocolo (o servicio) que envió este segmento
Puerto destino:	es el protocolo (o servicio) que recibirá este segmento
Número de secuencia:	es la posición en el flujo de byte del emisor
Número de reconocimiento:	es el número del byte que la fuente espera recibir en el próximo segmento
Hlen:	es un número entero que especifica la longitud del encabezado del segmento
Bits de código (C Bits):	detalla el propósito y contenido del segmento
Ventana:	especificación de cuántos datos TCP están deseando aceptar
Suma de verificación:	número entero usado para verificar el encabezado TCP y la integridad de los datos
Puntero urgente:	campo para indicar que este segmento debe ser procesado de inmediato
Opciones:	detalles de negociación del tamaño máximo del segmento
Datos:	protocolos de alto nivel o información específica de la aplicación

Bits de código

Estos bits indican el tipo de segmento TCP y cómo debería ser procesado. La tabla 3.5 muestra los códigos establecidos y sus significados asociados. Estos códigos son análogos al campo tipo en el marco Ethernet, el cual significa que los segmentos TCP se identifican por sí mismos.

Tabla 3.5 Tipos de segmento TCP.

Códigos	Significado
URG	Puntero urgente (*Urgent pointer*)
ACK	Reconocimiento (*Acknowledgment*)
PSH	Solicita un empujón (*Request a push*)
RST	Reiniciar (*Reset*)
SYN	Sincroniza los números de secuencia y empieza la conexión
FIN	Finalización de la corriente de bytes

Ventana

TCP posee un elaborado mecanismo para manejar buffer de datos, control de flujo y retransmisión de segmentos no reconocidos. El campo ventana ayuda al TCP a determinar cuántos datos está dispuesto a aceptar en el siguiente segmento. El tamaño de los datos de una transacción puede tener un impacto significativo sobre toda la red y el rendimiento de las aplicaciones. Para entender el por qué, supóngase por el momento que se ha establecido una conexión TCP entre dos nodos llamados `socrates` y `durer`, y que durante las transacciones anteriores, `socrates` le ha especificado a `durer` una ventana TCP de 1024 (la cual es la predeterminada). Ahora, `durer` comienza a experimentar un alto uso y empieza a correr bajo en recursos disponibles, como la memoria. Muchas razones pueden causar esta situación. En este momento, `socrates` todavía le está enviando mensajes TCP a `durer`, pero `durer` tiene problemas en reconocer los segmentos (o tal vez hasta en procesarlos) de `socrates`, debido al número de mensajes. Como `durer` tiene problemas de recursos, el siguiente segmento enviado por `socrates` contiene un tamaño de ventana menor, el cual informa a `socrates` que debe ajustar la cantidad de datos contenidos en los mensajes TCP subsiguientes. Después de que `socrates` recibe el nuevo tamaño de ventana, empieza a enviarle a `durer` cantidades más pequeñas de datos.

Una vez que la limitación de recurso ha sido resuelta en durer, ya sea por acción explícita de parte del administrador del sistema o por la finalización de las tareas que causaron el problema en primer lugar, durer le envía a socrates un tamaño de ventana mayor y reanuda el procesamiento tal como antes. Sin la habilidad del TCP para ajustar dinámicamente el tamaño de los segmentos, en el ejemplo, durer empezaría a abandonar los mensajes que no pueda procesar. Esto, en consecuencia, haría que socrates los retransmitiera, desperdiciando no sólo ciclos de procesamiento en socrates, sino el ancho de banda debido a los mensajes retransmitidos.

Puntero urgente

Ya que TCP suministra una interfaz de flujo, es importante a veces que una aplicación tenga una forma de enviar un mensaje fuera-de-banda o urgente al otro extremo de la conexión, sin tener que esperar que los mensajes anteriores sean procesados. Un ejemplo de por qué un fuera de banda es importante, se presenta cuando un usuario desea terminar una sesión de conexión remota usando Telnet. Frecuentemente, los terminales proporcionan interrupciones o señales de control, las cuales pueden usarse para informarles a las aplicaciones que deberían finalizar. En este caso, TCP usa el bit URG para indicar que éste es un segmento fuera de banda, y establece el puntero urgente para especificar la posición en el segmento donde termina el dato urgente.

Opciones TCP

Este campo negocia el tamaño del segmento TCP, el cual es útil en una situación donde es posible establecer una unidad máxima de transferencia (MTU) mayor o menor. Los valores MTU pueden ser diferentes en distintas redes físicas. Por ejemplo, ATM tiene una MTU mayor que Ethernet.

Capa Internet

La capa Internet (o red del modelo OSI) suministra un servicio de entrega que no es confiable, y está basada en un protocolo de transferencia no orientado a la conexión. Tal como se indicó previamente, el protocolo de Internet (IP) opera en esta capa suministrando un servicio de transferencia de primera clase, y es responsable de enrutar los paquetes de diferentes redes IP. Los paquetes IP pueden perderse, demorarse, ser duplicados y entregados en desorden.

Se han definido dos versiones del protocolo. La versión implementada más ampliamente es la 4 (conocida como IPv4) y, debido a las deficiencias del protocolo y las limitaciones de recursos de esta versión, se realizaron ampliaciones que dieron como resultado una nueva versión conocida como IPv6. Sin embargo, la versión 6 no se ha implementado ampliamente en la industria de trabajo en red.

Las principales características y servicios del IP (versión 4) son:

- entrega no confiable
- protocolo no orientado a la conexión
- viaje del paquete por diferentes rutas
- formato de dirección
- subred
- enrutado

Entrega no confiable

La expresión *no confiable* indica que el IP no hace el intento de garantizar la entrega de un paquete a su destino. Esto está en agudo contraste con el comportamiento y los servicios del protocolo de control de la transmisión, los cuales suministran una facilidad de transferencia confiable que asegura la entrega del mensaje. El IP, por otro lado, proporciona una facilidad de entrega de primera clase y no asegura la transferencia del paquete, pero tampoco lo descarta caprichosamente. A pesar del hecho de que el IP no es confiable, no significa que los datos llevados en IP no vayan a ser entregados de manera correcta. El IP simplemente usa un protocolo de nivel superior como el TCP, para asegurar la entrega garantizada de mensajes.

Protocolo no orientado a la conexión

Se dice que el IP no es orientado a la conexión porque no establece una conexión por medio de la cual transfiera paquetes, lo cual es opuesto al comportamiento de los protocolos de transferencia confiables. La entrega del paquete está basada en la información de la dirección IP contenida en el mismo paquete. Cada paquete IP es autónomo, independiente de cualquier otro paquete,

y no hace parte de un acuerdo preestablecido entre las entidades de la red. Debido a que no se mantiene ninguna información de conexión en el IP, la entrega del paquete es simplificada y más eficiente.

Paquetes en rutas diferentes

Con el IP, los paquetes pueden viajar por diferentes rutas para alcanzar su destino final, aunque cada paquete pueda cargar una porción más pequeña de un mensaje mucho mayor. Este comportamiento se observa cuando los paquetes de IP viajan en Internet. Igualmente, los paquetes podrían llegar en desorden.

Direccionamiento IP

El IP define el formato de la dirección y requiere que cada entidad de red tenga su propia y única dirección. Las direcciones contienen un par: la red y la identificación de nodo, que se expresan como un solo número. Con el IPv4, se usan 32-bits para representar una dirección IP y se expresan en notación punteada. Cada dirección se escribe como cuatro enteros decimales separados por puntos decimales. Se han definido cinco clases diferentes en el IPv4. Sin embargo, en la práctica, sólo las tres primeras clases se usan para definir un par de red/nodo, como se muestra en la figura 3.10.

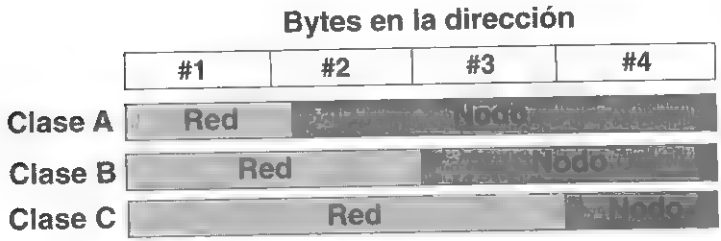


Figura 3.10 Las tres primeras clases principales de dirección IP.

Cada clase especifica el formato utilizado para interpretar cuánto se usa de la dirección para representar la red y cuánto se usa de la dirección para representar el nodo. Las interpretaciones de las direcciones incluyen:

Clase A: el primer byte es la identificación de la red, y los demás bytes especifican el nodo. El rango de dirección de la red (primer byte) es 1-127.

Clase B: los primeros dos bytes son la identificación de la red, y los demás bytes son el nodo. El rango de dirección de la red es 128-191.

Clase C: los primeros tres bytes son la identificación de la red, y el byte restante es el nodo. El rango de dirección de la red es 192-223.

Dos clases adicionales (D y E) están definidas, pero están reservadas y no pueden usarse para direcciones de red normales. Las direcciones de clase D se emplean para un apoyo de multidifusión, y las direcciones clase E están reservadas para uso futuro. Una forma de distinguir las diferentes clases consiste en usar la regla del primer byte. Con esta regla, el primer byte determina a qué clase pertenece la dirección. Por ejemplo, el uso de la dirección IP 10.1.3.2,10 es el primer byte de esta dirección. El número 10 cae en el rango de 1-127; por tanto, esta dirección IP es clase A y la porción de red es 10; mientras la porción del nodo es 1.3.2.

El IP también define algunas direcciones reservadas que incluyen direcciones de ciclo de retorno (*loop back*) y de difusión. La red de ciclo de retorno está definida como la dirección 127 y se usa como una red privada para diagnósticos y soporte internos de dispositivos IP. Esta dirección de red es reservada y no se debe usar como una dirección de red genuina. De hecho, las especificaciones del protocolo IP no recomiendan su uso en una red activa. La dirección de ciclo de retorno puede observarse usando el comando UNIX **ifconfig –a**. La dirección de difusión definida como 255 también se considera especial, ya que denota un atajo para direccionar todos los *hosts* en un rango dado. Por ejemplo, dada una red de 134.110.0.0, la cual es una red clase B, la dirección de difusión de 134.110.255.255 direcciona todos los dispositivos en toda la red 134.110. Debido al significado especial asociado con 255, no debe usarse como dirección de nodo.

La asignación de las direcciones IP se cumple a través de una agencia central conocida como Centro de Información de Red (*Network Information Center,* NIC). El NIC es responsable de asignar direcciones de red IP únicas para cualquier organización que desee conectarse Internet. En muchos casos, un proveedor de servicios de Internet local (ISP) le solicitará una dirección IP en su nombre o le proporcionará una suya.

Subred

Subred es un mecanismo usado para dividir una red en subredes más pequeñas o *subredes*. Una gran motivación para implementar las subredes consiste en distribuir el control administrativo de asignación de la dirección IP. Las subredes también permiten un uso más efectivo de las direcciones existentes. Con las subredes, la porción del nodo de la dirección IP se divide en dos secciones: la dirección de la subred y la dirección del nodo, como se muestra en la figura 3.11.

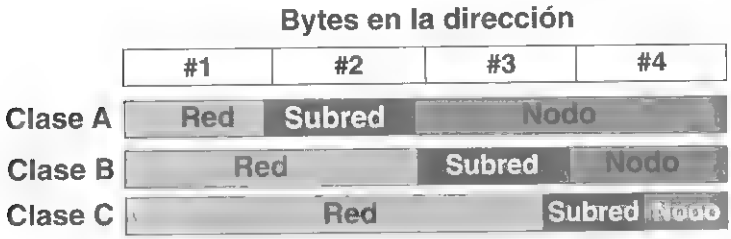


Figura 3.11 Direccionamiento de las subredes.

Para implementar la subred, se deben cumplir los siguientes requisitos. Primero, se debe crear una máscara de subred para usarla en cada uno de los dispositivos que participarán en la subred. Esta máscara de subred es una dirección especial de 32-bits, la cual se expresa como una dirección IP normal, usando una notación decimal punteada. Como en el caso de la dirección IP regular, cada uno de los octetos en la subred está en el rango de 1 a 255. Pero a diferencia de las direcciones IP, los octetos representan un grupo de bits enmascarados que se combinan con la dirección IP del dispositivo para darle prioridad a la red de la subred. En particular, determinar la subred involucra combinar la máscara de subred y la dirección IP del *host* con el operador Booleano and. La figura 3.12 muestra el cálculo de subred requerido.

Segundo, cada dispositivo que va a participar en una subred debe usar la misma dirección de la máscara de subred. En particular, para cada interfaz definida en el sistema local, la máscara de subred debe ser definida junto con los otros parámetros de la interfaz. Por ejemplo, bajo UNIX, el comando **ifconfig** se invoca

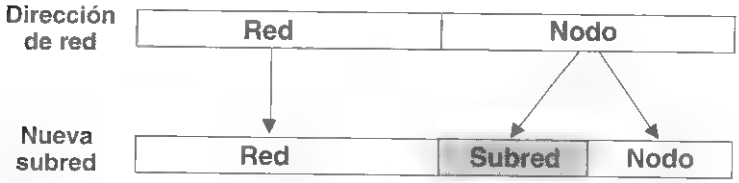


Figura 3.12 Cálculo de la subred.

en cada interfaz del sistema instalada en el sistema. La información de la máscara de subred debe incluirse cuando se ejecute este comando del sistema.

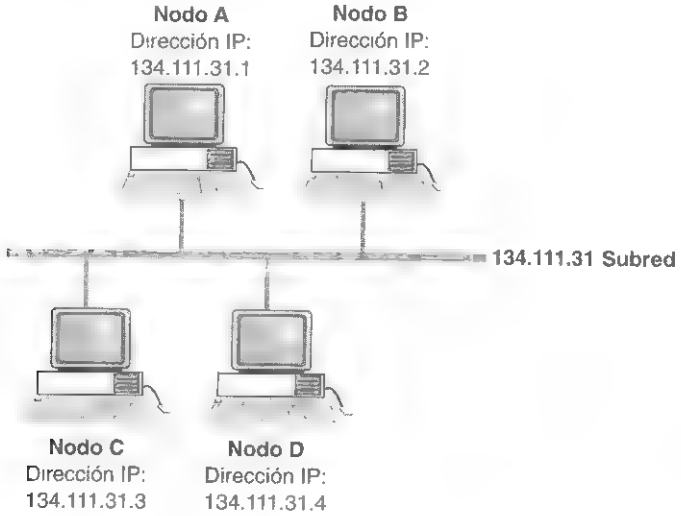


Figura 3.13 Subred de muestra.

Para ilustrar un poco más la implementación de una subred, consideremos el ejemplo de red que se muestra en la figura 3.13. Esta figura presenta cuatro dispositivos unidos a la misma red. Cada uno de los dispositivos ya ha recibido una dirección IP, pero ahora debemos determinar cuál máscara de subred se debe usar. Hemos determinado que la red crecerá en el futuro y necesitará tener suficientes direcciones IP en la subred para aproximadamente 200 dispositivos. Como resultado, necesitaremos usar por lo menos 8 bits (1 octeto) para la subred.

Las direcciones IP que se usan en la red del ejemplo son de clase B. Recuerde que las direcciones clase B están en el rango de

128-191 y los primeros dos octetos son la dirección de red (134.111), mientras los dos restantes (31.1, por ejemplo) son direcciones de nodo. El único lugar del que podemos tomar bits para usarlos en la subred es el de la porción del *host* de la dirección. Simplemente, no podemos usar ninguna de las direcciones de red ya que esa área de la dirección es restringida. Teniendo esto en cuenta, necesitamos subdividir la porción *host* en dos, que es donde la máscara de subred entra en juego. Usando todos los unos (1) en los bits de los campos de la subred y los ceros (0) en los bits del campo del nodo, podemos forjar la dirección de subred deseada convirtiendo la dirección IP y la máscara de subred en un binario:

```
(decimal)          (binario)
134.111.31.1       10000110.01101111. (Dirección IP)
                   00011111.00000001
255.255.255.0      11111111.11111111. (Máscara de subred)
                   11111111.00000000
134.111.31.0       10000110.01101111. (Dirección de subred)
                   00011111.00000000
```

Aplicar el operador booleano `and` generará la dirección de subred de `134.111.31.0`. Después de que esta máscara de subred haya sido configurada en todos los dispositivos en la subred, todos los nodos harán el cálculo `and` requerido para determinar la dirección de subred a la cual pertenecen. En muchos (si no en todos) sistemas UNIX, ésta es una tarea automática cuando se inician los servicios de red del sistema. En resumen, para subdividir una red, deben seguirse los siguientes pasos:

1. Determinar la clase de dirección IP de la red que quiere subdividir.
2. Determinar la cantidad de direcciones que se requiere para la subred.
3. Determinar si se puede usar un octeto completo (por lo menos con una dirección clase B) para la dirección de la subred.
4. Realizar `and` booleano en una de las direcciones del dispositivo usando la máscara de subred.
5. Aplicar la máscara de subred a todos los dispositivos que participarán en la subred.

La asignación de la máscara de subred en este ejemplo fue relativamente sencilla, ya que se usó un octeto completo para el número de subred. Esto también hizo que el cálculo and fuera muy sencillo. Sin embargo, suponga que cambiamos el ejemplo, usando en su lugar un número de red clase C. Las direcciones que ahora necesitamos asignar a la red del ejemplo en la figura 3.13 incluyen 199.81.23.1 hasta 199.81.23.4. Con una dirección clase C, el rango del primer octeto se encuentra entre 192 223. Los primeros tres octetos son la dirección de red (199.81.23), mientras el octeto restante (4, en este ejemplo) es la dirección del nodo. Este ejemplo complicará el proceso de subdividir la red un poco, ya que debemos saber separar la porción del *host* y la dirección de subred de sólo un octeto. Sin embargo, se aplica el mismo procedimiento anterior.

Para poder formular una dirección de subred desde una dirección de red clase C, debemos tomar prestado algún número de bits de la porción del *host* de la dirección para representar el número de subred. Al hacerlo, reducimos automáticamente el número de direcciones IP que estarán libres para cada dispositivo en la subred. Por tanto, necesitamos saber exactamente cuántos dispositivos se instalarán en la subred. Para perfeccionar un poco más nuestro ejemplo, suponga que sólo necesitaremos diez direcciones IP para nodos individuales en la subred.

Podemos determinar la máscara de subred y otra información que esté disponible, pero es mucho más fácil y rápido si usamos un programa de software que haga los cálculos por nosotros. Un calculador excelente está disponible en el sitio Web de Chesapeake (www.cci.com/tools/subcal). Este software proporciona la habilidad de subdividir cualquier dirección IP válida y nos muestra los cálculos y pasos involucrados. La figura 3.14 muestra la pantalla principal. Dése cuenta de que este software puede bajarse y correr desde una computadora local o ejecutarse directamente desde el sitio de Chesapeake. Las instrucciones de descarga e instalación para el uso en una computadora local se encuentran en el apéndice C.

El primer campo, Dirección del Host IP (*IP Host Address*), necesita que especifiquemos la dirección IP de uno de los *hosts* en nuestra red. En este caso, le damos una de las direcciones de nodo de nuestro segundo ejemplo (es decir, 199.81.23.1). Luego, le damos algo de información acerca de la subred ingresando información en el tercer cuadro, con la leyenda "Select One of the Following". Podemos hacer esto en una

de cuatro formas: especificando directamente la máscara de subred (campo `Subnet Mask`), el número de bits para la subred (campo de `Bits`), el número máximo de *hosts* en la subred (campo `Maximum Possible Subnets`), o el número de posibles *hosts* por subred (campo `Possible Hosts per Subnet`). El cambio de uno de estos campos cambia los otros tres automáticamente. Por ejemplo, cuando una máscara de subred se cambia a `255.255.255.224`, el número de bits se convierte en tres y el número de posibles *hosts* por subred llega a 30.

La relación entres estos campos se puede ver más fácilmente observando la ventana de cálculos que se muestra en la figura 3.15. Esta pantalla contiene cinco secciones. La primera muestra

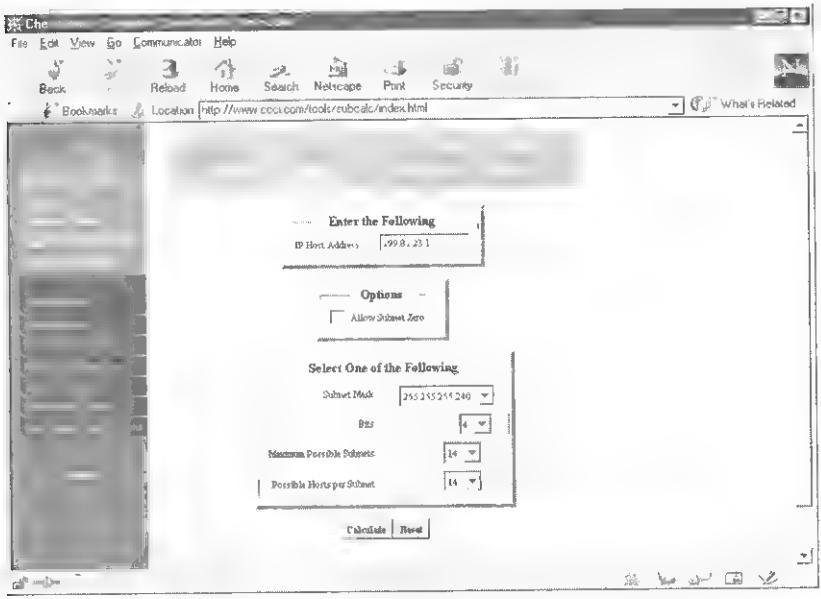


Figura 3.14 Ventana principal de cálculo de la subred.

dos líneas; cada una contiene tres campos, la clase de Dirección IP estándar (campo `Class C Network`) en formato decimal y binario, los campos de bits de `Subnet` y `Host`, respectivamente, y la máscara de subred. Ésta es la máscara que se debe aplicar a cada sistema que se incluirá en la subred. Esta máscara de subred usa 28 bits y puede ser referenciada como una "máscara de subred de 28 bits". La segunda sección contiene el número de

subred de la operación lógica AND de los campos de dirección del *host* (`199.81.21.1`) y la máscara de subred (`255.255.255.224`). La tercera muestra la máscara de subred correcta, la mascara de subred invertida y el número de subred expresado en binarios. La siguiente sección contiene la dirección dirigida de subred y la dirección de difusión. La dirección de difusión dirige los paquetes IP a cada nodo en la subred local. La sección final incluye el resumen que contiene la dirección de subred, las direcciones *host* disponibles especificadas con un rango, y la dirección de difusión dirigida. El rango disponible del *host* es importante, ya que representa las direcciones que pueden usarse por todos los sistemas *host* en la subred.

Protocolo de mensaje de control de Internet

Como se analizó previamente, IP proporciona un mecanismo de entrega de paquete de primera clase que no garantiza la entrega de datos. IP es aislado de los detalles de protocolo de más bajo nivel y hace pocas suposiciones acerca del rendimiento y confiabilidad de la red. La entrega de datos se cumple sin ninguna coordinación especial entre las otras entidades IP y, en su mayoría, opera bastante bien. Sin embargo, pueden existir condiciones más allá del control de IP que hacen difícil o imposible la entrega. Por ejemplo, un *host* remoto podría ser removido temporalmente de la red o un enrutador podría sobrecargarse cuando de arranque una aplicación intensiva de red. IP necesita una forma de detectar este tipo de problemas.

El protocolo de mensajes de control de Internet (*Internet Control Message Protocol,* ICMP) ofrece una facilidad de mensaje de error general para el protocolo IP y los servicios relacionados. ICMP proporciona una forma para que las entidades IP que encuentren errores puedan reportarlos a la fuente que causó el problema. La entidad de la fuente es responsable de corregir el error. Esto podría incluir, por ejemplo, reportar el error a una aplicación de nivel mayor. La tabla 3.6 nombra algunos de los tipos de mensaje ICMP.

Una de las utilidades de depuración del trabajo en red más popular, `ping`, utiliza ICMP para determinar el alcance del *host*. `ping` utiliza las primitivas de solicitud eco (*echo*) y respuesta que están disponibles en ICMP, para determinar si un *host* remoto está disponible y opera correctamente en la red. La figura 3.16 muestra la operación básica de la solicitud y la respuesta de eco.

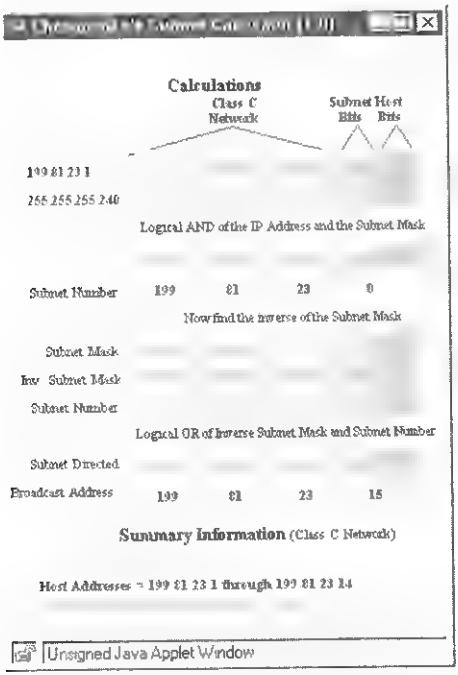


Figura 3.15 Ventana principal de cálculo de subred.

Tabla 3.6 Tipos de mensaje de error ICMP.

Tipo de mensaje	Significado
Solicitud/Respuesta de eco	Determina el alcance del sistema
Destino inalcanzable	No se puede alcanzar el destino deseado
Retirar fuente	Detener el envío de datos
Redirigir	Detección de un error de enrutado
Tiempo excedido	Paquete IP caducado

Un usuario quiere determinar si el nodo B está disponible en la red. Desde el nodo A, el usuario entra un comando **ping** que emite una solicitud de eco al nodo B. Si el nodo B está activo, le responde al nodo A usando una respuesta de eco. Si el nodo B no

está activo, entonces el `nodo A` no recibe respuesta. En este caso, la solicitud del `nodo A` simplemente queda extemporánea. Cuando esto ocurre, le aparece al usuario un mensaje extemporáneo que indica que el `nodo B` no respondió a la solicitud.

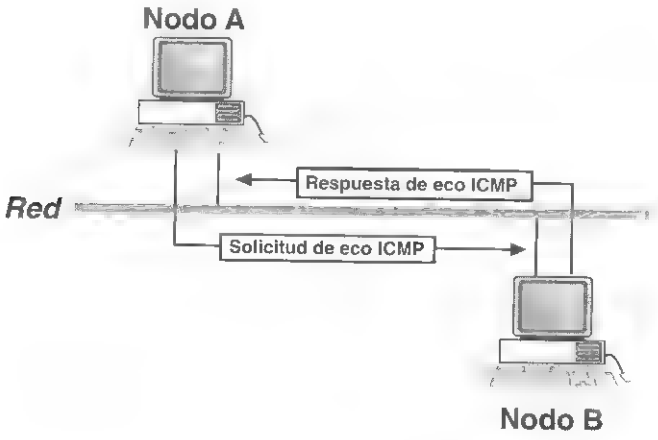


Figura 3.16 Solicitud/respuesta de eco ICMP.

Destino inalcanzable

Cuando un enrutador no puede entregar o enviar un paquete a su destino, emitirá al remitente un mensaje de *destino inalcanzable* ICMP. Los errores de destino inalcanzable generalmente implican algún tipo de problema o error de enrutado (es decir, ruta incorrecta en una tabla de enrutado). Los destinos pueden ser inalcanzables debido a fallas en el hardware que hacen que un sistema no esté temporalmente disponible, o cuando un remitente especifica una dirección de destino no existente. A veces es imposible saber que un destino es inalcanzable, a menos que se use la utilidad del sistema `ping`. Por ejemplo, si un usuario emite una solicitud Telnet a un sistema remoto (el cual se encuentra conectado a un LAN Ethernet) y el sistema ha sido desconectado de la red, esta solicitud Telnet eventualmente será extemporánea, debido a que Ehernet no soporta ningún marco de reconocimiento. Como resultado, no se generarán mensajes de destino inalcanzable, ya que Ethernet supone la entrega de datos aun para un nodo que haya sido desconectado temporalmente de la red.

Retirar la fuente

Retirar la fuente en ICMP suministra una forma de manejar la congestión del paquete IP. La congestión en una red puede ser causada por varios factores. Sin embargo, una causa principal se presenta cuando varios sistemas comienzan una transmisión de datos que fluye a través de un solo enrutador, como se muestra en la figura 3.17. En este caso, el enrutador mismo es sobrecargado por el tráfico combinado. Otra causa de congestión se presenta cuando un sistema de computadora poderoso transmite datos a un sistema menos poderoso que no puede seguirle el paso. Si estas situaciones continúan sin revisión, el enrutador y el sistema de bajo poder eventualmente empezarán a descartar los paquetes, lo que en consecuencia hará que el TCP los vuelva a transmitir, empeorando así la situación. En este caso, el enrutador y el sistema pequeño simplemente emiten un mensaje de retiro de fuente, lo que le indica al emisor que debe parar de enviar datos. Así, el enrutador y el sistema más pequeño tienen tiempo para procesar cualesquiera paquetes que estén pendientes.

No hay una forma directa de revertir los efectos del mensaje de retiro de fuente ICMP. Sin embargo, cualquier destino que reciba un retiro de fuente disminuye la tasa a la cual transmite paquetes al destino. El sistema remoto empieza a incrementar gradualmente la tasa de transmisión luego de no recibir mensajes de retiro de fuente adicionales, y de manera eventual continúa el tráfico normal.

Desviación

El mensaje de desviación de ICMP informa a los enrutadores de los cambios en la información de rutas en la red. Una de las suposiciones básicas con el TCP/IP es que los enrutadores conocen la topología de las rutas en una red y esta información la comparten los enrutadores participantes. Cuando un enrutador detecta un *host* que usa una ruta no óptima, emite un mensaje de desviación informándole al *host* que hay una ruta mejor disponible. Es decisión del *host* incorporar esta nueva información en su configuración de rutas. En algunos casos esto no es automático y debe ser hecho por el administrador de red. La figura 3.18 muestra un ejemplo de dónde un enrutador enviará un mensaje de desviación ICMP. Suponga que el `nodo A` quiere hablar con el `nodo D`. En este caso, existen dos vías entre el `nodo A` y el `nodo D`, llamadas `R1` y `R2`. La vía `R1` es más corta y contiene

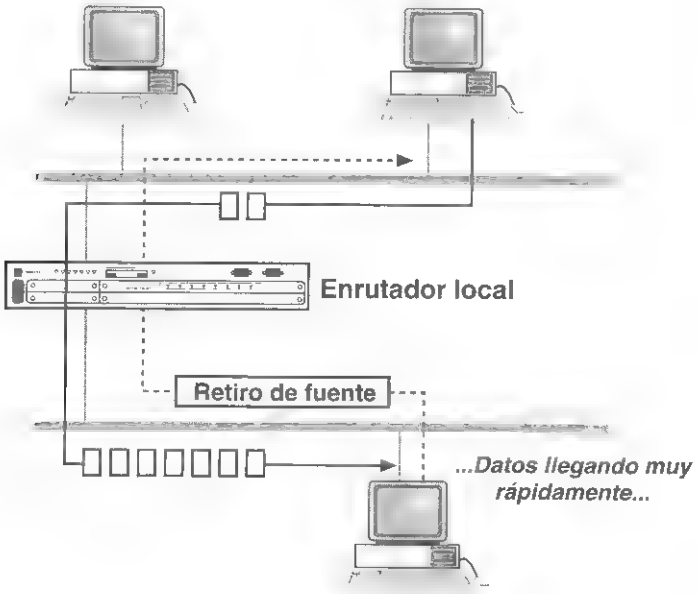


Figura 3.17 Desviación del retiro de fuente ICMP.

menos saltos entre el `nodo A` y el `nodo D`. La vía `R2` contiene menos uniones extra, ya que sólo suministra conectividad a las otras redes que incluyen el `nodo B` y el `nodo C`. El `nodo A` debe intentar comunicarse con el `nodo D` por medio de `R2`, luego `R2` enviará el mensaje de desviación de vuelta al `nodo A` diciéndole que debe usar `R1`, la cual tiene una ruta más corta al `nodo D`. Muchas veces, la causa de usar un enrutador incorrecto se encuentra en la configuración del mismo sistema o en la información incorrecta de enrutado emitida desde uno o más enrutadores mal configurados. En cualquiera de los dos eventos, las desviaciones ICMP ayudan al administrador de red a localizar este tipo de problemas.

Los mensajes de desviación no proporcionan un mecanismo común para distribuir la información de enrutado, ya que están limitados a la interacción entre los anfitriones (*hosts*) y los enrutadores. En su lugar, los protocolos de enrutado, como el protocolo de información de enrutado (*Routing Information Protocol*, RIP), se usan para propagar la información de enrutado en una red.

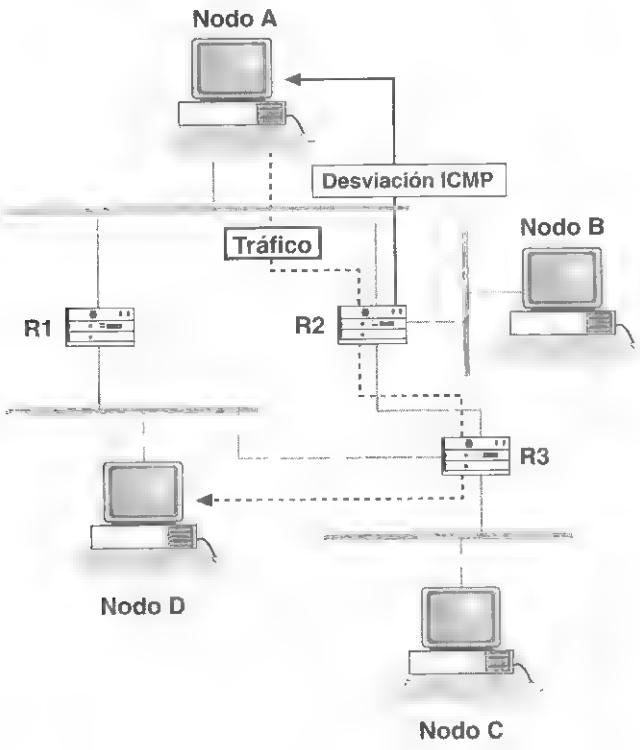


Figura 3.18 Desviación ICMP.

Tiempo excedido

Cada paquete IP contiene un campo de tiempo de vida, que controla el tiempo durante el cual puede existir un paquete en la red. Cuando un enrutador procesa el paquete, disminuye este campo antes de enviarlo. Siempre que un enrutador descarta un paquete debido a que su campo de tiempo de vida es cero o debido a que algún otro está fuera de tiempo, el enrutador emite un mensaje de *tiempo excedido* ICMP de vuelta a la fuente del paquete. Éste es el primer mecanismo usado para detectar ciclos de enrutado o rutas excesivamente largas en una red. El comando **traceroute**, por ejemplo, usa el campo de tiempo de vida para trazar la ruta que un paquete IP puede tomar entre dos puntos, y también mide la cantidad de tiempo requerido para recorrer cada ruta por esta vía hasta el destino.

Protocolo de resolución de dirección

Como ya se analizó, IP impone cierta estructura y formato en las direcciones usadas por los dispositivos de trabajo en red. En la versión 4 de IP, la dirección es de 32 bits de largo y está expresada en notación decimal punteada. En la versión 6, la dirección es aún más larga, 128 bits. Como en el caso de cualquier protocolo de red de mayor nivel como el IP, existe un requisito para que un mecanismo transforme estas direcciones en direcciones usadas por dichos protocolos como Ethernet, FDDI o TokenRing. Conocidos como protocolos de enlace de datos (*datalink*), tienen su propia estructura y formato de dirección, los cuales son bastante diferentes del IP. Por ejemplo, Ethernet usa direcciones de 48 bits y se expresa usando la notación hexadecimal con dos puntos. Así, `8:0:20:4:cf:2c` representa una dirección Ethernet válida. En este caso, esta dirección de enlace de datos no siempre se transforma fácilmente con las direcciones usadas por IP, lo cual también es válido para muchos otros protocolos de enlace de datos. Esto crea un dilema, ya que sin una forma de transformar las direcciones IP a interfaces físicas, la comunicación entre los nodos no sería posible.

Es conveniente y bastante necesario que esté disponible una forma simple, flexible y poderosa de transformación entre las direcciones IP y de enlace de datos (físicos). El protocolo de resolución de direcciones (*Address Resolution Protocol*, ARP) ha probado ser una forma muy refinada y sólida de resolver el problema de transformación de direcciones. En particular, el ARP se diseñó para manejar la resolución de la dirección de cualquier protocolo de capa de red al protocolo de enlace de datos, no sólo entre IP y Ethernet.

La operación básica de ARP es simple: cuando el `nodo A` quiere determinar la dirección física del `nodo B`, el `nodo A` enviará un mensaje de difusión especial solicitando la dirección del `nodo B` a todos los dispositivos en la red, como lo muestra la figura 3.19. Todos los *hosts* reciben el mensaje, incluidos el `nodo B` y el `nodo C`, pero sólo el `nodo B` responderá con la dirección física correcta, como lo muestra la figura 3.20. La razón por la que sólo el `nodo B` responde es que éste examina el paquete ARP y determina que su propia dirección IP coincide con la dirección IP objetivo que está buscando el solicitante.

Una función hermana del ARP, conocida como protocolo de resolución de dirección al contrario (*Reverse Address Resolution Protocol*), o RARP, hace la inversión; dada una dirección de enlace de datos, encuentra la dirección IP correspondiente. RARP

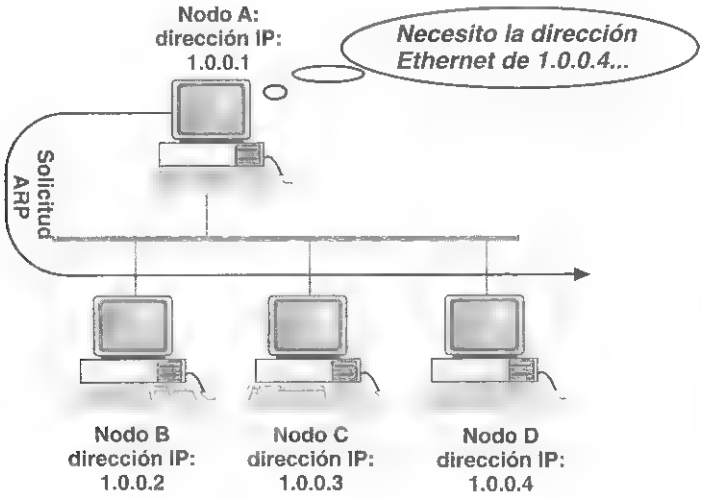


Figura 3.19 Solicitud ARP.

se usa principalmente cuando estaciones de trabajo sin disco necesitan determinar sus direcciones IP durante el inicio de un sistema.

El término *difusión* (*broadcast*) en el ejemplo anterior indica un paquete dirigido localmente a todos los dispositivos de red en un segmento LAN en particular. Ciertos protocolos de enlace de datos como Ethernet ofrecen la facilidad de transmitir los paquetes de difusión a todas las estaciones conectadas usando una dirección de destino especial. Con Ethernet, todos los 1s (o FF en hexadecimal) en la dirección de destino se consideran una difusión. ARP supone que los detalles de obtener mensajes para cada estación serán manejados directamente por el protocolo de enlace de datos.

Como se puede observar, el protocolo ARP es muy simple. Sin embargo, a pesar de su simplicidad, tiene unas cuantas ventajas de fondo. Primero, ARP es de naturaleza dinámica, lo que evita la necesidad de rastrear de manera estática y almacenar IP para direcciones físicas. Este comportamiento de resolución dinámica libera al administrador de la red de mantener y asegurar constantemente la información correcta cada vez que se añade o se retira un dispositivo de la red. Con las redes actuales, sería imposible cumplir dicha tarea, dado el tamaño y la tasa de crecimiento de muchas internets. Segundo, ARP es estándar y está

disponible en todos los dispositivos que soporten TCP/IP, independientemente del protocolo de enlace de datos. Es por eso, por ejemplo, que IP puede operar en FDDI y TokenRing. Esto hace que la construcción de las redes heterogéneas sea mucho más fácil, ya que ARP oculta los detalles de direccionamiento del enlace de datos, y el diseñador de redes no necesita darle importancia a la resolución de la dirección física cuando se combina con diferentes protocolos de enlace de datos. Tercero, ARP puede usarse en subredes IP para proporcionar a los dispositivos una forma muy simple de comunicarse en una Internet enrutada.

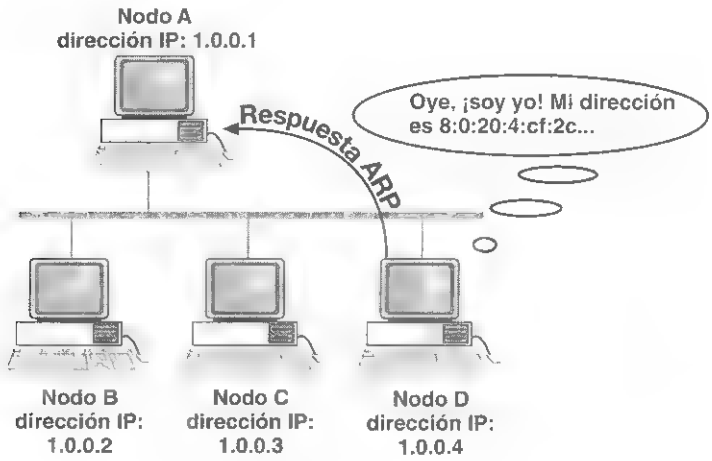


Figura 3.20 Respuesta ARP.

Formato del paquete

A diferencia de los otros protocolos de trabajo en red, los campos del paquete ARP no son todos de formato fijo, sino que se apoyan en ciertos campos fijos al principio del paquete. ARP y RARP comparten el mismo formato de paquete básico. Para hacer que el ARP sea útil para otros protocolos de enlace de datos, la longitud de los campos dependerá del tipo de red. La figura 3.21 muestra el formato de ARP estándar y el del mensaje RARP.

El campo de *tipo de hardware* (*hardware type*) define el tipo de interfaz del hardware para el cual el emisor requiere respuesta. Los valores estándar incluyen 1 para Ehertnet y 2 para FDDI. El campo del *protocolo* especifica el tipo de protocolo de alto nivel que el emisor está usando, el cual, para IP, es 0800 en

hexadecimal (en decimal es 2048). El campo de *operación* nota si el mensaje es una solicitud ARP (1), una respuesta ARP (2), una solicitud RARP (3) o una respuesta RARP (4). Los campos *Hlen* y *Plen* definen el hardware y las longitudes de campo de dirección del protocolo de alto nivel. Los campos *Hlen* y *Plen* permiten que ARP trabaje con cualquier protocolo de enlace de datos al azar, ya que los tamaños de los campos del *tipo de hardware* y de *protocolo* pueden determinarse inspeccionando el paquete ARP directamente. La entidad emisora proporcionará información, en lo posible, del campo del *emisor HA* y/o del *emisor IP* cuando se emita la solicitud. El *emisor HA* se usa con la solicitud ARP, mientras el *emisor IP* se usa con una solicitud RARP. El *target HA* se usa cuando se responde un mensaje ARP; por el contrario, el *target IP* se usa cuando se le responde a RARP.

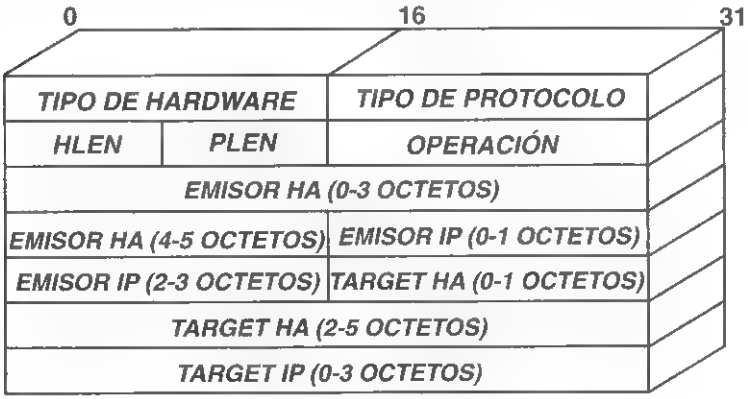


Figura 3.21 Formato de mensaje ARP/RARP.

Caché ARP

A simple vista, el servicio ARP puede parecer ineficiente, ya que enviará un paquete de difusión cada vez que un dispositivo quiera comunicarse con otro. Sin embargo, las implementaciones ARP, en muchos sistemas, incluyen el uso de un caché ARP para almacenar temporalmente vínculos a direcciones de solicitudes ARP anteriores. De hecho, antes de que se haga una solicitud ARP, el software de más alto nivel examina el caché ARP para ver si ya existe en el caché la transformación de la dirección de enlace de datos requerida. Si es así, se usa el vínculo existente; si, por el contrario, no existe, se envía la solicitud ARP. Como resultado del caché, el tráfico de red se reduce, ya que las solicitudes ARP sólo

se hacen cuando los dispositivos no son conocidos. Igualmente, el rendimiento de la aplicación mejora, ya que la solicitud ARP del emisor puede satisfacerse usando el caché, en lugar de transmitir un paquete en la red. Por último, el caché le proporciona al administrador una forma de ver la conectividad entre los protocolos de más alto nivel y el hardware de la red. Como se verá más adelante en este libro, inspeccionar el caché ARP puede ser una forma eficaz de resolver problemas de la red y del sistema.

Aquí se presenta una interesante pregunta en cuanto al caché ARP: ¿Cuándo se deben desechar del caché los vínculos? La respuesta no es simple. Si queremos mantener el caché ARP poblado de entradas eternamente (o hasta que el sistema sea reiniciado), existe la posibilidad de que el caché contenga información no válida. Considérese, por ejemplo, una red que contiene los sistemas `monet` y `rembrandt`, que están en medio de la comunicación cuando la tarjeta de interfaz de red (*network interface card*, NIC) falla en `rembrandt` y es, por consiguiente, remplazada. La nueva NIC contendrá una nueva dirección de enlace de datos, a menos que el administrador la haya cambiado después de que la interfaz se haya instalado. En consecuencia, `monet` ya no puede seguir hablando con `rembrandt`, ya que el caché en `monet` contiene una dirección de enlace de datos no válida para `rembrandt`. Por otro lado, podemos tomar el enfoque de que los vínculos en el caché ARP deben expirar en un periodo relativamente corto, por ejemplo cada 30 segundos. Desafortunadamente, esto afectará de manera adversa el rendimiento de la red, ya que se necesitarán más difusiones de la red porque el caché ARP será muy pequeño. Sin embargo, resolverá el problema de los vínculos incorrectos de NIC que se han remplazado recientemente, debido a que la vieja entrada habría sido depurada en un periodo razonable después de que la nueva NIC fuera operacional.

Quizás la mejor solución a este problema puede ser descrita tomando *el enfoque del camino del medio*. Es decir, la expiración del vínculo no debe ser tan pequeña que sea inefectiva ni demasiado grande para abordar los cambios en la red. En general, muchos sistemas UNIX borrarán entradas ARP en 20 minutos aproximadamente. Algunas versiones de UNIX también permiten que los administradores cambien la extemporaneidad del caché ARP para satisfacer:

- el formato de direcciones de enlace de datos (*datalink*)
- los requisitos de red individual y, cuando sea necesario, alterarlos para abordar otros problemas relacionados con la red.

Formato de la dirección de enlace de datos

Tal como se indicó, las direcciones de enlace de datos se expresan en 48 bits (seis bytes) y están separadas por dos puntos (:). Esta notación de dos puntos se usa como el primer método para representar estas direcciones de hardware. Algunas herramientas de administración de la red UNIX, como los comandos **ifconfig** y **arp**, usan este formato. Las direcciones de protocolo de enlace de datos para FDDI, Ethernet y TokenRing contienen un número de identificación de código del vendedor y un número de serie. Esta información puede usarse en la identificación del nodo, para fines de inventario. La autoridad del registro de IEEE le asigna la porción del vendedor a aquellas organizaciones que producen hardware de trabajo en red y solicitan un código de vendedor. Estos códigos también son conocidos como identificadores únicos de organización (*Organization Unique Identifiers*, OUI). Los tres primeros bytes de la dirección representan el fabricante o vendedor del dispositivo, y los otros tres bytes representan el número de serie de la unidad, como se especifica en la figura 3.22.

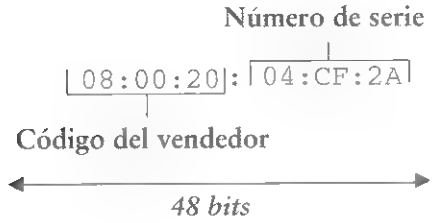


Figura 3.22 Formato de la dirección de enlace de datos.

Frecuentemente, en un sistema *multi-homed*, las direcciones de enlace de datos serán compartidas entre todas las interfaces definidas. Esto puede ser cierto también para sistemas que contienen diferentes tipos de interface de red (es decir, Ethernet y FDDI). La tabla 3.7 muestra algunos de los códigos de vendedor más comunes. Note que más de un código de vendedor puede estar asociado con el mismo vendedor. Esto puede ser cierto, por ejemplo, cuando una compañía compra otra compañía y sus productos. En este caso, el código del vendedor de la compañía que fue comprada se incluye en el de la otra compañía. Es por eso que la tabla 3.7 contiene múltiples entradas para 3Com y Cisco. Conocer unas cuantas de estas direcciones puede ser útil durante la

depuración de la red, para ayudar a identificar el tipo de dispositivo y el vendedor al cual pertenece un paquete determinado. Si usted está interesado en hallar un vendedor que no está ennumerado en la tabla 3.7, ingrese a la página de Internet de IEEE en *http://standards.ieee.org/regauth/oui/oui.txt*.

Tabla 3.7 Muestra de códigos OUI de vendedor.

Código de vendedor	Vendedor
00:20:AF 00-80-3E 00-C0-D4 (muchos más)	3Com Corporation Anteriormente Synernetics, Inc. Anteriormente AXON, Inc.
08-00-2B AA-00-00 AA 00 03 AA-00-04	Anteriormente Digital Equipment Corporation (DEC), ahora es propiedad de Cabletron Systems
00:00:0C 00-E0-B0 00-E0-F7 00-E0-F9 00-E0-FE 00-06-7C 00-06-C1 00-10-07 (muchos más)	Cisco Systems, Inc. (Incluye una amplia lista de antiguas compañías)
00-60-69	Brocade Corporation
00-00-81 00-00-A2 00-80-E3	Anteriormente Bay Network, Inc. Anteriormente Coral Networks
00-E0-B1	Packet Engines, Inc.
08:00:5A	IBM Corporation
08:00:20	Sun Microsystems, Inc.
08-00-09	HP Corporation
00-10-E3	Compaq Corporation

Al acceder a este localizador uniforme de recursos (*Universal Resource Locator,* URL) se le presentará todo el archivo OUI. Note que este archivo es bastante grande (más de 900K) ¡y está creciendo! Si no está interesado en ver todos los contenidos del archivo OUI, busque los códigos de vendedor específicos que está buscando ingresando a *http://standards.ieee.org/regauth/oui/index.html.*

Al ingresar a este URL se le presentará la página que se muestra en la figura 3.23.

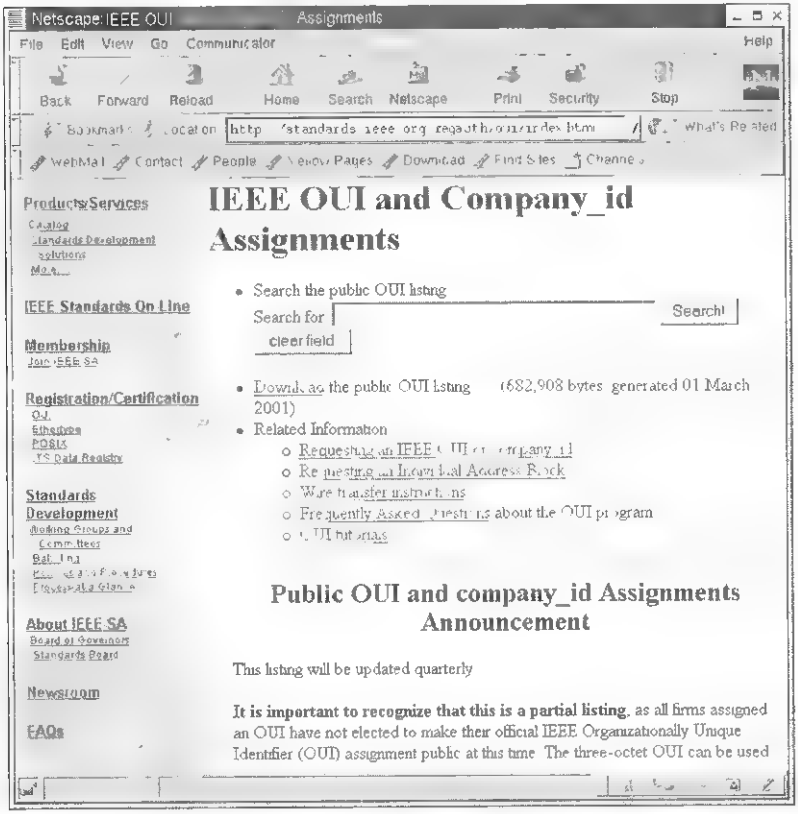


Figura 3.23 Página de búsqueda de OUI de IEEE.

Si quiere buscar por vendedor, ingrese el nombre de la compañía en el campo de búsqueda. Además, si usted conoce el OUI, pero quiere el vendedor, entre la dirección en su lugar. Después, haga click en el botón *Search!* La figura 3.24 muestra el resultado que se obtuvo cuando se buscó la compañía Brocade.

Figura 3.24 Resultados de búsqueda de OUI.

04

Herramientas y utilidades centrales del sistema

Este capítulo proporciona una perspectiva general de los comandos y las utilidades disponibles en el sistema operativo Linux. Muchas de las herramientas analizadas son parte de un conjunto común de herramientas y están disponibles en muchas otras plataformas UNIX. Por ejemplo, el comando **arp** opera de una manera similar y tiene muchas de las mismas opciones de línea de comandos en los sistemas Solaris, HPUX y AIX.

El software que se analiza en esta sección incluye herramientas que pueden ayudar al administrador de red a manejar las tareas relacionadas con la red y abordar problemas de red específicos. Este capítulo cubre las siguientes herramientas:

Comando	Página
arp	110
ifconfig	119
netstat	129
ping	147
tcpdump	158
traceroute	177

Herramienta ARP

Descripción general

El comando **arp** presenta y manipula la tabla de direcciones de red en un sistema UNIX local. La tabla del protocolo de resolución de direcciones, conocida como el *caché ARP*, contiene una lista completa de todas las transformaciones de protocolo de enlace de datos a direcciones IP para la red local. Recuérdese que en el capítulo 3 se mencionó que el protocolo ARP es una facilidad dinámica que transforma direcciones de enlace de datos como Ethernet a direcciones IP. Siempre que un sistema quiere transmitir un mensaje, debe primero conocer la dirección de bajo nivel (es decir, de enlace de datos) para el sistema objetivo. Muchas herramientas de trabajo en red, como `telnet`, `ftp` y otras, usan indirectamente la tabla ARP.

El comando **arp** proporciona la capacidad de ver y modificar el caché ARP. Nótese que las expresiones tabla ARP y caché ARP pueden usarse de manera intercambiable y se refieren a lo mismo. Igualmente, los términos vinculación (*binding*) y transformación (*mapping*) se refieren a una entrada ARP. Los usuarios regulares de UNIX (usuarios diferentes de *root*) pueden presentar el caché ARP pero no pueden hacer cambios de ningún tipo. La razón es obvia: los usuarios podrían alterar el caché, lo que afectaría la conectividad de todos ellos.

Con el comando **arp**, el superusuario puede:

- presentar el caché ARP
- borrar una entrada ARP
- añadir una entrada ARP

La herramienta ARP proporciona un número de opciones de línea de comandos. Sin embargo, sólo las más importantes se enumeran en la tabla 4.1 y se describen totalmente más adelante. Cada una de estas opciones soporta un solo carácter de comando o palabra clave, junto con dos guiones al principio. Por ejemplo, la opción `-v` tiene una clave **--verbose** asociada. Éstas son equivalentes y pueden usarse de manera intercambiable en la línea de comandos. La tabla 4.1 muestra ambas formas. La razón para escoger una en vez de la otra es simplemente cuestión de estilo. Sin embargo, el formato de la palabra clave puede facilitar el recordar las opciones individuales en el futuro.

Tabla 4.1 Opciones importantes de línea de comandos **arp**.

Opción	Palabra clave	Descripción
`-a`	**--display**	Presenta la entrada ARP actual para un *host* específico
`-d`	**--delete**	Borra una entrada ARP
`-f`	**--file**	Carga un archivo que contiene entradas para poner en el caché
`-i`	**--device**	Presenta sólo aquellas entradas para una interfaz especificada
`-n`	**--numeric**	Muestra direcciones numéricas, en lugar de nombres de *hosts*
`-s`	**--set**	Crea una entrada ARP
`v`	**--verbose**	Presenta el caché ARP en el modo expresivo *(verbose)*

Presentación del caché ARP

Para presentar los contenidos de la tabla de ARP, se usa el comando **arp -v** (**--verbose**). Cuando se presenta la tabla, ésta incluye el nombre del dispositivo, el tipo de hardware, la dirección física,

la bandera de máscara y la interfaz de la red. Una muestra de esto es la siguiente:

```
# arp -v
Address              HWtype    HWaddress           Flags Mask  Iface
durer.home.com       etner     00:60:97:0E:A3:06   C           eth0
switch.home.com      ether     00:00:1D:0A:5D:01   C           eth0
probe.home.com       ether     00:C0:D4:00:09:40   C           eth0
rembrandt.home.comether        08:00:20:82:be:05   C           eth0
rubens.home.com                     (incomplete)               eth0
monet.home.com       ether     08:00:20:04:cf:2c   C           eth0
Entries: 6 Skipped: 0          Found: 6
```

El campo `Address` muestra el nombre del *host* o la dirección IP que corresponde a la entrada ARP. Ésta es la dirección que se usa para buscar la tabla ARP, con el fin de determinar si existe la entrada deseada. Si se presenta una dirección IP (en lugar de un nombre de *host*), se debe concluir que la dirección IP no pudo ser resuelta correctamente para un nombre de *host* o que se utilizó la opción `-n`. Esto podría ser el resultado de una entrada no válida o que no existe en el servidor DNS o en el archivo `/etc/hosts`, una mala configuración en cualquiera de ellos, o problemas de conectividad con el servidor DNS.

El campo `HWType` representa la interfaz de red para la cual se obtuvo la entrada ARP. Como se puede observar, la palabra **ether** representa entradas que se derivan de una red Ethernet. También tienen soporte otros tipos de red adicionales, incluidos ARCnet (`arcnet`), PROnet (`pronet`), AX.25 (`ax25`) y NET/ROM (`netrom`).

El campo `Flags Mask` (banderas de máscara) suministra detalles adicionales relacionados con la entrada ARP. La tabla 4.2 muestra los indicadores o las banderas disponibles y sus significados.

El campo `HWaddress` (dirección física) contiene la dirección de bajo nivel para cada nodo en la red local, y se expresa con

Tabla 4.2 Indicadores de entrada ARP.

Bandera ARP	Significado
C	entrada ARP se ha completado
M	entrada ARP permanente (estática)
P	publicar esta entrada ARP

seis números hexadecimales con notación de dos puntos. Recuérdese que la notación de dos puntos usa el número hexadecimal separado por dos puntos, como en

```
00:10:5a:28:5a:7c
```

El campo `iface` corresponde a la interfaz de red que está conectada a la red local, para la cual se obtuvo la entrada ARP. En el ejemplo anterior, se muestran dos interfaces separadas con sus propios listados. En la práctica, la mayoría de sistemas contienen sólo una interfaz única. `eth` significa Ethernet, que es el nombre del controlador del dispositivo de hardware Ethernet. El cero al final indica que es la primera interfaz definida en el sistema.

Las entradas ARP que están marcadas con `P` son permanentes (o estáticas) y fueron definidas fuera del protocolo ARP normal (por ejemplo, se ingresaron manualmente con el comando **arp -s**). Las entradas estáticas se usan también para soportar sistemas de trabajo en red que no soportan el protocolo ARP. Por tanto, la dirección de vínculo se debe entrar manualmente en la tabla ARP. Un problema con este enfoque es que si se cambia la dirección IP o la dirección de hardware de cualquiera de estas transformaciones del sistema, la información ARP se debe actualizar manualmente para reflejar el cambio. La bandera `C` indica una entrada normal y completa, y en su mayoría, si no todas, las entradas ARP deben estar marcadas con este indicador.

ARP sólo advierte las direcciones IP asociadas con las interfaces genuinas. Es interesante notar que la información ARP que se obtiene de otros dispositivos en red no es normalmente publicada por sistemas de terceros. Generalmente, es responsabilidad de cada dispositivo responder a las solicitudes ARP con su propia información. Hay un caso especial, en el que un sistema dará respuesta a solicitudes ARP en nombre de otros. Se conoce como *proxy ARP*, y se usa con subredes enrutadas u otras configuraciones de red especiales.

Las entradas ARP transformadas incluyen un temporizador, que controla cuánto tiempo permanecerá la entrada en el caché ARP. Por defecto, muchos sistemas UNIX mantienen la entrada durante aproximadamente 20 minutos antes de retirarla. Algunos sistemas permiten que el administrador altere este valor de tiempo.

La cadena `incomplete` indica que una solicitud ARP todavía está pendiente y la transferencia no se ha resuelto para el *host* asociado. En el ejemplo ARP anterior, al sistema `rubens` le falta

una dirección física, ya que la solicitud ARP no obtuvo respuesta. Esto se conoce como una entrada ARP *incompleta*. Muchos factores podrían causar esta condición: problemas del hardware con el dispositivo, problemas de cableado de la red o problemas de configuración del software en el *host*. Lo esencial es que el software ARP no pudo comunicarse con el dispositivo para obtener la información deseada. La cadena final que se presenta muestra un conteo del número de entradas en el caché.

En situaciones donde hay muchas entradas ARP, puede usarse `-a` seguido de un nombre de *host* válido para enumerar un sistema específico. Por ejemplo, el siguiente comando enumerará la transformación ARP para el *host* llamado `socrates`:

```
# arp -a socrates
socrates.nome.com (10.0.2.201) at 00:60:97:0E:A3:03 [ether] on etn0
```

Igualmente, se pudo haber usado la dirección IP, en lugar del nombre del *host*. Así, el comando **arp -a 10.0.2.201** generaría la misma salida.

Nota

Con la versión estándar Linux 6.0, la página manual para el comando **arp** establece de manera incorrecta que se puede usar la opción `-display` en lugar de la opción `-a`. Desafortunadamente, el programa ARP no soporta esto, y el usuario obtendrá un error cuando intente usar la opción `-display`. Así, el comando **arp --display socrates** producirá:

```
arp: incorrect number of options
```

El comando **arp** también acepta nombres de *host* o direcciones IP sin la opción `-a`. Sin esta opción de comando, cada una de las etiquetas de campo se muestra junto con la entrada ARP. Así, el comando

```
# arp socrates
Address           HWtype  HWaddress          Flags Mask  Iface
socrates.home.com ether   00:60:97:0E:A3:03      C      eth0
```

proporciona más información que cuando se usa la opción `-a`. Otra opción que ARP soporta es `-n`. Esta opción presenta nombres de *host* o direcciones IP en los listados. Una razón para

presentar las entradas ARP usando las direcciones IP es que es un poco más rápido. La transformación entre las direcciones IP y los nombres de *host* la hace por lo general el servidor de nombres de dominio (*Domain Name Server*, DNS) y, como resultado, puede tomar algo de tiempo para completarse. En la práctica, la demora es insignificante en muchas situaciones. Sin embargo, si la aplicación de resolución del nombre (NIS o DNS, por ejemplo) experimenta demoras o algún otro problema operacional, enumerar el caché ARP con la opción `-n` será mucho más rápido. El comando **arp -n** devolverá la salida de menera más rápida si la resolución del nombre es no funcional por completo, dado un caché ARP moderadamente grande. En el primer ejemplo dado en esta sección, el formato de los listados ARP presentará lo siguiente:

```
# arp -n -v
Address          HWtype  HWaddress          Flags Mask   Iface
10.0.2.10        ether   00:60:97:0E:A3:06      C         eth0
10.0.2.60        ether   00:00:1D:0A:5D:01      C         eth0
10.0.2.50        ether   00:C0:D4:00:09:40      C         eth0
10.0.2.127       ether   08:00:20:82:be:05      C         eth0
10.0.2.220               (incomplete)                     eth0
10.0.2.126       ether   08:00:20:04:cf:2c      C         eth0
Entries: 6 Skipped: 0    Found: 6
```

Como se puede observar, se presenta la dirección IP de cada *host* en lugar de los nombres reales del *host*.

Previamente se mencionó que el comando **arp** presenta el nombre de la interfaz para la cual se obtuvo la entrada. Dada la salida anterior, toda la transformación ocurrió desde la interfaz `eth0`. Algunas veces, un sistema puede contener más de una interfaz. Esto se conoce como *multi-homing*. Cuando el caché ARP se presenta en estos sistemas, la salida contiene entradas para cada una de las interfaces, como en el siguiente ejemplo:

```
Address              HWtype  HWaddress          Flags Mask  Iface
durer.home.com       ether   00:60:97:0E:A3:06      C        eth0
moster.home.com      ether   00:60:97:0A:B3:09      C        eth1
probe2.home.com      ether   00:80:96:0B:C1:01      C        eth1
switch.home.com      ether   00:00:1D:0A:5D:01      C        eth0
probe.home.com       ether   00:C0:D4:00:09:40      C        eth0
```

Podemos apreciar que existe transformación para dos interfaces en este sistema. En este caso, `eth0` y `eth1` se enumeran en la salida. Como ARP rastrea entradas por interfaz, podemos usar la opción `-i` para presentar sólo las transformaciones desde una

interfaz en particular. El siguiente comando

```
#arp -i eth1
```

mostrará todos los listados ARP para la interfaz `eth1`:

```
Address              Wtype   HWaddress           Flags Mask   Iface
moster.home.com      ether   00:60:97:0A:B3:09       C        eth1
probe2.nome.com      ether   00:80:96:0B:C1:01       C        eth1
```

Borrado de un caché ARP

Podría ser necesario borrar una o más entradas de la tabla ARP. Por ejemplo, si una falla en el hardware pudiera dar como resultado el remplazo de una tarjeta de interfaz de red, cambiaría la dirección del hardware de red del sistema. En este caso, la entrada ARP existente no reflejará que la dirección de bajo nivel cambió. Como consecuencia, los mensajes enviados a este *host* desde otros dispositivos no serán recogidos, debido a que la dirección de hardware utilizada no concuerda con la de la interfaz.

A fin de resolver este problema, debe usarse la opción `-d` para borrar una entrada ARP, como se muestra más adelante. Debido a que remover los vínculos ARP puede causar problemas en la red, sólo se le permite removerlos al superusuario. El comando **arp** espera que se use la opción `-d` con un *host* o una dirección IP válido. En este ejemplo, el *host* `durer` se remueve de la tabla ARP:

```
# arp -d durer
```

Si la tabla ARP se presenta en este momento, la entrada del `durer` se nombraría como aparece aquí:

```
Address              HWtype    HWaddress       Flags Mask     Iface
durer.home.com                 (incomplete)
```

Si un usuario diferente de *root* intenta remover este *host* de la tabla, se presentaría un mensaje de error:

```
% arp -d durer
SIOCDARP(priv): Operation not permitted
```

Agregado de una entrada de caché ARP

Varias condiciones pueden justificar añadir manualmente entradas a la tabla ARP. Una situación como ésta ocurre cuando se necesita una comunicación con un dispositivo, pero el dispositivo, por alguna razón, no soporta ARP, o la implementación no

es funcional. Éste podría ser el caso de un sistema muy antiguo que todavía está en servicio. Además, si una dirección NIC cambia, la tabla debe actualizarse manualmente para asegurar la conectividad. Finalmente, puede ser necesario agregar entradas para soportar los servicios ARP *proxy*.

Para agregar una entrada ARP, se usa la opción `-s` seguida por el nombre del *host* (o dirección) y la dirección de enlace de datos física. Por ejemplo, supongamos que quisiéramos agregar un sistema llamado `bruegel` a la tabla ARP. El formato del enlace de datos físico se representa como `x:x:x:x:x:x`, donde cada `x` es un número hexadecimal entre `0` y `FF`. Para ilustrar el ejemplo, se usaría el siguiente comando:

```
# arp -s bruegel 08:00:20:82:be:05
```

Si no se muestra ningún mensaje de error, puede suponerse que el comando tuvo éxito. También se puede enumerar el caché ARP para comprobar si `bruegel` se agregó en realidad:

```
# arp bruegel
Address           HWtype  HWaddress           Flags Mask  Iface
bruegel.home.com  ether   08:00:20:82:be:05     CM        eth
```

Nótese que este vínculo ARP es etiquetado con la bandera `M`, que indica que es una entrada permanente, ya que fue agregada en la línea de comandos. Como las transformaciones permanentes se definen estáticamente, no se consideran extemporáneas. Desafortunadamente, las entradas permanentes no se guardan con el reinicio del sistema.

Para entender por qué modificar la tabla ARP está restringido sólo al superusuario, debe tenerse en cuenta que cuando se definen las entradas ARP estáticas, no existe ninguna unión entre esta información y los dispositivos reales conectados a la red. Por ejemplo, la entrada para `bruegel` se agregó manualmente, pero no hay un mecanismo disponible en el ARP que asegure que la dirección física es en realidad la correcta. El comando **arp** supone que la información suministrada es exacta y que el dispositivo está conectado a la red. Por tanto, si los usuarios normales tuvieran la capacidad de modificar la tabla ARP, probablemente se introducirían más errores y éstos llevarían a una pérdida de la conectividad de la red. Si la dirección física de un recurso de red crítico (como un servidor) se cambia de manera incorrecta, se detendría toda comunicación entre el sistema local y el sistema crítico.

Carga de vínculos ARP usando un archivo

El comando **arp** también soporta la capacidad de agregar vínculos que son definidos en un archivo regular, en lugar de especificar la información en la línea de comandos. Este método se usa, por ejemplo, para cargar una lista de entradas ARP al tiempo, en un solo paso. No es poco común, por ejemplo, para una serie de vínculos ARP, que sean cargados durante los procedimientos normales de inicio del sistema, especialmente si éste no soporta esquemas que consulten bases de datos distribuidas, como el NIS. El formato de un archivo utilizado con este propósito incluye los siguientes campos:

```
hostname datalink address [ temp ] [ pub ] [netmask]
```

Usando un editor de texto, se puede crear una lista de vínculos ARP en el formato anterior y guardarlo como un archivo de texto; por ejemplo, `/etc/arp list`. A continuación se muestra listado del ejemplo del archivo:

```
bruegel 08:00:20:82:be:05 temp
rubens 08:00:20:81:ce:01 temp
rembrandt 08:00:20:86:fe:02
cezanne 08:00:20:81:bb:01
michelangelo 08:00:20:84:ee:02
```

Nótese que tanto `bruegel` como `rubens` contienen la palabra clave **temp**, la cual indica que estas entradas serán borradas de la tabla ARP luego de que expiren. En la práctica, la opción `temp` no tiene un uso generalizado, ya que podría no ser deseable que estos vínculos expiren y sean removidos del caché. La palabra clave **pub** anuncia el vínculo ARP a la red.

El comando **arp** puede usarse para soportar servicios *proxy* ARP. El *proxy* ARP es una forma de responder a las solicitudes ARP en nombre de otro dispositivo de red. Esta configuración se utiliza comúnmente para soportar dispositivos detrás de un enrutador o una configuración especializada, como las que se requieren para las estrategias de acceso telefónico remoto. La opción `netmask` se usa principalmente para habilitar los servicios ARP *proxy* para uno o más sistemas en una subred. Como los servicios *proxy* normalmente no se implementan usando una configuración ARP estática, la opción `netmask` se considera obsoleta.

El archivo contiene las entradas deseadas. La opción `-f` del comando **arp** se utiliza para cargar los contenidos del archivo dentro de la tabla ARP:

```
# arp -f /etc/arp-list
```

Si se quiere cargar esta tabla automáticamente, se edita uno de los archivos de inicio del sistema (`/etc/rc.d/init.d/network`, por ejemplo) y se incluye:

Listado 4.1 Secuencia de comandos de inicio ARP.

```
1   #!/bin/bash
2   # Load custom ARP table
3   #
4   echo -n "Loading local arp table..."
5   if [ -f /etc/arp-list ]; then
6   /sbin/arp -f /etc/arp-list
7   echo "done."
8   else
9   echo "ARP file not found!"
10  exit -1
11  fi
```

Cuando se agrega esta secuencia de comandos, a una secuencia de comandos de inicio de red, cargará las transformaciones ARP contenidas en el archivo `/etc/arp-list`. La secuencia de comandos verifica si el archivo `/etc/arp list` está presente y si es un archivo regular (línea 5). Si lo es, invocará al comando **arp** con la opción `-f`. Si no, presenta un mensaje de error y luego sale con un error.

Herramienta ifconfig

Descripción general

El comando **ifconfig** es la abreviatura de *configuración de la interfaz*, y se usa para configurar las interfaces de red local. Este comando se invoca normalmente durante el inicio del sistema para configurar cada interfaz con una configuración predefinida. El comando **ifconfig** puede usarse también para cambiar los parámetros de interfaz de la red después de que el sistema haya sido iniciado. Este comando se usa generalmente para depuración y/o actividades de afinación del sistema, y realiza las siguientes funciones:

- enumera la configuración de cada interfaz de red definida
- inhabilita/habilita cualquier interfaz de red definida
- modifica los parámetros de configuración de la interfaz de red
- crea seudointerfaces

El comando **ifconfig** proporciona un gran número de opciones y palabras clave que se utilizan para configurar una o más interfaces. La tabla 4.3 enumera las opciones y palabras clave más importantes, que se usan como modificadores con otras opciones de línea de comandos.

Listado de interfaces disponibles

Para presentar todas las interfaces del sistema, se usa el comando **ifconfig -a**.

Tabla 4.3 Opciones y palabras clave **ifconfig**.

Opción	Descripción
`-a`	Se aplica a todas las interfaces que estén instaladas actualmente en el sistema
`arp` `-arp`	Habilita el uso del protocolo de resolución de direcciones (ARP) en esta interfaz; inhabilita el uso del ARP con la opción `-arp`
`promisc` `-promiscatento`	Habilita el modo promiscuo (es decir, está atento a todo el tráfico) en esta interfaz. La opción `-promisc` lo inhabilita.
`allmulti` `-allmulti`	Habilita todo el tráfico de multidifusión que vaya a recibir la interfaz. La opción `-allmulti` inhabilita la recepción de paquetes de multidifusión
`broadcast` `-broadcast`	Con el argumento dado, establece o borra la dirección de difusión para esta interfaz. Ésta es una dirección de difusión de la capa de red
`pointtopoint` `-pointtopoint`	Habilita un modo punto a punto para esta interfaz. La suposición básica es que es un enlace dedicado entre dos dispositivos. Si se proporciona el argumento de dirección, establece la dirección de protocolo para el otro lado del enlace. Inhabilita el modo *point-to-point* con la palabra clave **pointtopoint**
`up`	Hace que la interfaz suba o se active

Tabla 4.3 Opciones y palabras clave **ifconfig**. (*continuación*)

Opción	Descripción
`down`	Hace que la interfaz baje o se desactive
`netmask`	Establece la máscara de red IP para esta interfaz. El argumento especificado puede estar en la forma `255.0.0.0` (decimal punteado) o `0xff00000` (hexadecimal)
`broadcast`	Establece la difusión IP para esta interfaz. El argumento especificado puede expresarse en el mismo formato que la palabra clave **netmask**
`address`	Establece la dirección IP para esta interfaz. Ésta debe ser una dirección IP única que no se haya asignado previamente a otro sistema

```
#ifconfig -a
eth0  Link encap:Ethernet HWaddr 00:10:5A:28:5D:7C
      inet addr:10.0.2.201 Bcast:10.0.2.255
      Mask:255.255.255.0

UP BROADCAST RUNNING PROMISC MULTICAST MTU:1500 Metric:1
          RX packets:822 errors:1 dropped:0 overruns:0 frame:1
          TX packets:108 errors:0 dropped:0 overruns:0 carrier:0
         collisions:0 txqueuelen:100
         Interrupt:10 Base address:0xfc00

lo        Link encap:Local Loopback
          inet addr:127.0.0.1 Mask:255.0.0.0
          UP LOOPBACK RUNNING MTU:3924 Metric:1
          RX packets:92 errors:0 dropped:0 overruns:0 frame:0
          TX packets:92 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0

ppp0      Link encap:Point to-Point Protocol
          POINTOPOINT NOARP MULTICAST MTU:1500 Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:10
```

La opción `-a` indica que **ifconfig** debe presentar todas las interfaces instaladas en el sistema, sin tener en cuenta su configuración actual o el estado operacional.

El ejemplo anterior indica que están definidas tres interfaces. La primera, `eth0`, representa las interfaces físicas de hardware

para Ethernet de 10 Mbs usando el controlador Ethernet 3Com. La segunda, `lo`, es la interfaz de ciclo de retorno, que se utiliza principalmente para comunicación interna y diagnósticos. La interfaz de `loopback` (ciclo de retorno) puede usarse para determinar si el software TCP/IP está operando correctamente en el nivel local. Por ejemplo, es posible hacer un **ping** en la dirección de ciclo de retorno para determinar respuestas válidas. La interfaz final (`ppp0`) representa un enlace de marcación punto a punto para conectividad con un proveedor de servicios de Internet (ISP) local.

Para cada interfaz, la presentación incluye los siguientes campos:

`Link encap`	Este campo especifica el protocolo de encapsulamiento de enlace que la interfaz usará cuando transmita marcos de enlace de datos. Los tipos soportados incluyen Ethernet, Local Loopback y protocolo *point-to-point*.
`HWaddr`	Es la dirección de enlace de datos para el protocolo de encapsulamiento. Por ejemplo, Ethernet usa la notación hexadecimal, como en la entrada para la interfaz `eth0: 00:10:5A:28:5D:7C`.
`inet addr`	Es la dirección IP asociada con esta interfaz.
`Bcast`	Representa la dirección de difusión de la capa de red.
`Mask`	Representa la dirección de la máscara de subred.

Además, la presentación incluye los parámetros operacionales para la interfaz. Éstos incluyen `UP`, `BROADCAST`, `RUNNING`, `PROMISC` y `MULTICAST`. Estas opciones muestran el modo y el estado actual de la interfaz.

Luego, la presentación incluye campos que representan los contadores estadísticos, como paquetes recibidos (`RX packets`), paquetes transmitidos (`TX packets`), número de colisiones (`collisions`), y demás. Éstos suministran una manera relativamente fácil de hacer una prueba comparativa del desempeño de la interfaz. Por último, los campos restantes muestran el número de interrupción y la dirección Base de E/S del hardware de la interfaz. Nótese, también, que no todas las interfaces tienen un número de interrupción y dirección de E/S asociados. A la interfaz `loopback` le faltan estos campos, ya que no usa un hardware específico en el sistema.

Control del estado de administración

Con **ifconfig** es posible inhabilitar una interfaz activa o habilitar una interfaz inhabilitada mientras el sistema está corriendo. En el estado inhabilitado, no se permitirá el flujo de paquetes a través de una interfaz. Esto equivale a desconectar la interfaz de la red. Cuando una interfaz está inhabilitada, se considera que está abajo desde un punto de vista administrativo. Para poner una interfaz en estado abajo, se invoca el comando **ifconfig** con la interfaz apropiada y la opción de palabra clave **down**, tal como se muestra:

```
# ifconfig eth0 down
```

Se puede usar el comando **ifconfig** con el nombre de la interfaz, en lugar de la opción `-a` para enumerar una interfaz individual. El siguiente comando presenta la configuración de la interfaz que bajamos con el comando anterior.

```
# ifconfig eth0
eth0      Link encap:Ethernet HWaddr 08:00:20:04:CF:2C
          inet addr:10.0.3.127 Bcast:10.0.3.255
Mask:255.255.255.0
BROADCAST MULTICAST MTU:1500 Metric:1
RX packets:3452 errors:0 dropped:0 overruns:0 frame:0
TX packets:3212 errors:1 dropped:0 overruns:0 carrier:1
          collisions:0 txqueuelen:100
          Interrupt:38 Base address:0x3200
```

Nótese que las palabras clave **UP** y **Running** hacen falta en la salida. Así es como **ifconfig** indica que ha sido inhabilitada una interfaz y no está disponible para su uso. Para habilitar o activar esta interfaz, simplemente se usa la opción de comando **up**:

```
# ifconfig eth0 up
```

Cuando la interfaz está arriba, está disponible en la red y está recibiendo información de la red. Sin la indicación de la bandera **up**, la interfaz es operacional y quizás esté conectada a la red, pero ninguna información está fluyendo a la red o desde ella.

Modificación de los parámetros de la interfaz

Se pueden usar tres métodos para modificar los parámetros de la interfaz de red. Primero, empleando directamente **ifconfig**, se pueden hacer cambios en la línea de comandos, y tendrá efecto inmediato. El segundo enfoque incluye modificar el inicio del

sistema y/o los archivos del sistema que **ifconfig** usa para configurar la interfaz. Este enfoque asegura que los cambios de la interfaz se hacen permanentes y no desaparecerán entre reinicios del sistema. A veces se usan ambos enfoques. Con frecuencia, se debe implementar rápidamente un cambio de configuración antes de que el sistema pueda ser reiniciado en un tiempo conveniente. Como consecuencia, **ifconfig** puede usarse para hacer los cambios inmediatos. En general, cuando una nueva interfaz se instala en un sistema, esta configuración puede manejarse por el procedimiento de instalación. El tercer método comprende el uso de la aplicación panel de control Linux estándar, `control panel`, para hacer los cambios. El capítulo 9, "Herramientas habilitadas para la Web", describe esta aplicación de manera más detallada.

Usando **ifconfig**, puede cambiarse la siguiente información importante para una interfaz:

- dirección IP
- máscara de red
- dirección de difusión
- dirección de enlace de datos
- MTU

La dirección IP se especifica en la notación decimal punteada normal y representa la única dirección para el *host* en la red a la cual está conectado el sistema. La máscara de subred en la red (o máscara de red) especifica el filtro que se utiliza para calcular las porciones de red y el *host* para la subred. La dirección de difusión especifica la dirección IP a la cual deben ser dirigidos los paquetes de difusión. La dirección de enlace de datos representa la única dirección de hardware de bajo nivel usada por Ethernet o FDDI, y está asociada con el mismo hardware. La máxima unidad de transferencia (MTU) denota el tamaño máximo del mensaje que la interfaz puede manejar. El tamaño del mensaje estándar para Ethernet es 1500, para FDDI es 4096, y para TokenRing es 2048. La MTU es modificada rara vez y, cuando se modifica, no puede configurarse para soportar valores mayores de los que el hardware soportaría; sólo se pueden usar valores menores.

Supongamos que la dirección IP de un sistema debe cambiarse porque el sistema se ha movido a una subred diferente. En este

caso, la máscara de red y la información de difusión se mantienen iguales. Para hacer el movimiento a la nueva red basta con solamente cambiar la dirección IP de la interfaz. La antigua dirección IP es `128.197.9.10` y la nueva dirección IP es `128.197.10.1`. El siguiente comando se usaría para cambiar la información de la red sobre la marcha:

```
# ifconfig eth0 128.197.10.1
```

Para hacer que este cambio sea permanente, debemos modificar el archivo `/etc/hosts`. Este archivo contiene la conversión entre el nombre del *host* y la dirección IP asociada. En el inicio del sistema, la dirección IP se obtiene del archivo y se aplica a la interfaz. La máscara de red y la información de difusión son las mismas, podemos usar los valores existentes. El nombre del *host* puede especificarse igualmente en la línea de comandos, en lugar de la dirección IP. Así, el comando

```
# ifconfig eth0 fred
```

logra el mismo resultado, suponiendo que a `fred` le ha sido asignada la dirección IP `128.197.10.1`, ya sea en el archivo `/etc/hosts`, DNS, o en la base de datos NIS del *host*.

Como se puede observar, cambiar la dirección IP para una interfaz es relativamente sencillo. Sin embargo, cambiar otras características de la interfaz requiere un poco más de trabajo. Para ampliar el ejemplo anterior, supongamos ahora que debemos cambiar la máscara de red y la información de difusión. Para cambiar la interfaz sobre la marcha, deberíamos usar:

```
# ifconfig eth0 128.197.10.1 netmask 255.255.0.0 broadcast\
128.197.255.255
```

En el ejemplo anterior, las palabras clave **netmask** y **broadcast** deben usarse para identificar la información que sigue a cada palabra clave. Las máscaras de red contienen `1`s en las posiciones de los bits de la dirección de 32 bits que se van a usar para la red y la parte de subred (si es apropiado), y `0`s para la porción del *host*. La porción máscara de red/subred debe ocupar por lo menos tantos bits como sea el estándar para la clase particular de red. Si no se definen subredes, se usa la máscara de red estándar. Como se describe en el capítulo 3, "Conjunto de protocolos TCP/IP", el uso de las subredes exige que más bits, de los que normalmente se reservan para la porción *host* de la dirección, identifiquen la subred. Una máscara de red puede especificarse

de dos formas diferentes: notación decimal punteada y notación hexadecimal.

La notación decimal punteada se expresa en cuatro números de un solo byte separados por puntos (por ejemplo, `255.255.255.0`). El formato hexadecimal incluye usar el prefijo `0x` seguido de una cadena de valor hexadecimal. Por ejemplo, el valor hexadecimal para `255.255.255.0` es `0xffffff00`. Como **ifconfig** soporta ambos formatos, éstos se pueden intercambiar. Cada una de las clases de direcciones estándar IP tiene direcciones de máscara de red predeterminadas, como se muestra en la tabla 4.4. Nótese que estas direcciones son las estándar que se utilizan si no se implementan subredes. Las direcciones de máscaras de subred específicas que se utilizan en muchos sitios diferirán de éstas, ya que las subredes definidas usan más bits que el estándar para esa clase.

Tabla 4.4 Direcciones estándar de máscara de red IP.

Clase	Notación decimal punteada	Notación hexadecimal
A	255.0.0.0	0xff000000
B	255.255.0.0	0xffff0000
C	255.255.255.0	0xffffff00

La dirección de difusión puede especificarse de la misma manera que la dirección de máscara de red. Sin embargo, la dirección de difusión se forma generalmente cambiando todos los bits en la porción *host* de una dirección a 1s. Por ejemplo, la dirección de difusión para la red `128.197.0.0` es `128.197.255.255`.

Parámetros para configuraciones especiales

El comando **ifconfig** soporta los parámetros adicionales. Éstos incluyen:

- ARP
- multidifusión
- modo promiscuo
- tipo de medio (*media*)
- *point-to-point*

La palabra clave **arp** especifica que la interfaz debe soportar una resolución de dirección IP de tipo ARP. Cuando se crea una interfaz con **ifconfig**, lo predeterminado es que soporte ARP. Para inhabilitar el ARP en una interfaz, se usa la palabra clave **-arp**. En muchas redes, debe activarse ARP.

La palabra clave **allmulti** habilita o inhabilita (**-allmulti**) todos los modos de tráfico de multidifusión. Si están inhabilitados, los paquetes de multidifusión (es decir, paquetes con direcciones de red clase D) serán recibidos por la interfaz. A pesar del hecho de que el tráfico de multidifusión está disponible en la interfaz, una aplicación que soporte el tráfico de multidifusión necesitará que se esté corriendo para hacer uso de este tipo de tráfico. La multidifusión la utilizan aplicaciones de multimedia para transportar paquetes que contienen video en tiempo real y datos de audio.

La palabra clave **promisc** habilita la interfaz para que reciba todo el tráfico de red. Se conoce como *modo promiscuo* cuando todo el tráfico de red es leído, no sólo el tráfico normal enviado a éste por otros sistemas en la red. El comando **-promisc** se usa para inhabilitar este modo. Ciertas herramientas de trabajo en red, como **tcpdump**, habilitarán este modo automáticamente cuando estén en operación.

La palabra clave **media** cambia el tipo de conectividad física para la interfaz. No todas las interfaces soportan la capacidad de cambiar dinámicamente tipos de medio de la interfaz. Aquellas que lo hacen, pueden usar muchos de los tipos más comunes, como `10base2` para thin Etheernet, `10baseT` para Ethernet de par trenzado, y `AUI` para Ethernet externo asociado con Ethernet 10base5.

La palabra clave **pointtopoint** habilita el uso de un protocolo de encapsulamiento de capa de enlace *point-to-point*, que generalmente significa que existirá una conectividad directa entre dos sistemas. Se pueden usar los protocolos generalmente soportados, como PPP o SLIP.

Interfaces lógicas

El comando **ifconfig** crea y configura interfaces lógicas (también llamadas virtuales o seudointerfaces). Estas interfaces se comportan como interfaces físicas y se pueden usar para asignar direcciones IP múltiples al mismo sistema. Desde un punto de vista de configuración, las interfaces lógicas se configuran independientemente pero comparten la misma dirección física y las mismas características de interfaz que la interfaz física real.

Para configurar una seudointerfaz, se combina la interfaz física con un número de referencia de una interfaz lógica, separado por dos puntos. Por ejemplo, para configurar la primera interfaz lógica para `eth0`, puede usarse el siguiente comando:

```
# ifconfig eth1:1 10.0.2.128 netmask 0xffffff00 broadcast 10.0.2.255
```

Las interfaces lógicas son mostradas sólo como las físicas cuando se utiliza el comando **ifconfig -a.** La siguiente salida muestra una interfaz lógica definida desde la interfaz física `eth1`.

```
eth0      Link encap:Ethernet HWaddr 08:00:20:04:CF:2C
          inet addr:10.0.3.127 Bcast:255.255.255.255
            Mask:255.255.0.0
          UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:2 dropped:0 overruns:0 carrier:2
          collisions:0 txqueuelen:100
          Interrupt:38 Base address:0x3200

eth1      Link encap:Ethernet HWaddr 08:00:20:04:CF:2C
          inet addr:10.0.2.127 Bcast:10.0.2.255
            Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
          RX packets:1810 errors:0 dropped:0 overruns:0 frame:0
            TX packets:1173 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          Interrupt:55 Base address:0x3000

eth1:1    Link encap:Ethernet HWaddr 08:00:20:04:CF:2C
          inet addr:10.0.2.128 Bcast:10.0.2.255
          Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
          Interrupt:55 Base address:0x3000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1 Mask:255.0.0.0
          UP LOOPBACK RUNNING MTU:3924 Metric:1
          RX packets:360 errors:0 dropped:0 overruns:0 frame:0
           TX packets:360 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
```

Nótese que la seudointerfaz, `eth1:1`, contiene la misma dirección de hardware Ethernet (`08:00:20:04:CF:2C`) y el mismo nivel de interrupción (`55`) de la interfaz real. Éstas son claves adicionales que indican que esta interfaz es igual a la interfaz `eth1`.

Para remover una interfaz lógica, se usa la palabra clave **down.** Así, el comando

```
ifconfig -a eth1:1 down
```

removerá la interfaz `eth1:1` del sistema. Si esta interfaz lógica fue creada durante el inicio del sistema, entonces la interfaz será configurada de nuevo cuando el sistema se reinicie.

Herramienta netstat

Descripción general

El comando **netstat** proporciona una riqueza de información en cuanto al estado de las conexiones de red, información de enrutamiento y otros datos importantes relacionados con la red. Esta herramienta cuyo nombre es la abreviatura de *estado de red* (*network status*), es estrictamente para hacer seguimiento y es una de las ayudas de depuración más populares disponibles en UNIX. Diferentes opciones de línea de comandos controlan el comportamiento de la presentación de **netstat.** Dado esto, la funcionalidad puede dividirse en un número de categorías y usarse para hacer lo siguiente:

- enumerar las sesiones de red activa
- mostrar información y estadísticas de la interfaz
- presentar la información de la tabla de enrutamiento
- mostrar las estructuras de datos de la red

Esta herramienta también proporciona palabras clave específicas que controlan la operación y el formato de salida. La tabla 4.5 contiene las palabras clave más importantes que controlan qué estructuras de datos se presentan. Algunas de las opciones/palabras clave de comando tienen una opción de carácter único y una cadena nemotécnica. Por ejemplo, las opciones `-n` y `help`, las cuales presentan el resumen de ayuda de línea de comandos, pueden usarse de manera intercambiable.

Tabla 4.5 Palabras clave **netstat.**

Opción	Nemotecnia	Descripción
`-i`	`--interfaz`	Muestra los parámetros y la información estadística de la interfaz de la red
`-g`	`--groups`	Presenta la información de pertenencia al grupo de multidifusión

Tabla 4.5 Palabras clave **netstat**. (*continuación*)

Opción	Nemotecnia	Descripción
`-M`	`--masquerade`	Enumera todas las sesiones que usan las capacidades de enmascaramiento en FTP
`-N`	`--netlink`	Muestra las interfaces y la actividad del enlace en red
`-r`	`--route`	Muestra las tablas de enrutamiento de la red
`-t`	`--tcp`	Presenta las conexiones de socket TCP activo. La opción `-tcp` presentará continuamente estas conexiones hasta que sea interrumpida por el usuario

La tabla 4.6 contiene los modificadores de línea de comandos que proporcionan información adicional o modifican la salida cuando se usan con las opciones de palabras clave.

Tabla 4.6 Opciones del comando **netstat**.

Opción	Nemotecnia	Descripción
`-A`	`--af`	Especifica una familia de dirección diferente. Las palabras clave específicas incluyen **--unix**, **--ipx**, **--ax25**, **--netrom** y **--ddp**
`-c`	`--continue`	Hace que la salida sea presentada continuamente hasta que el usuario la interrumpa
`-h`	`--help`	Presenta al usuario una información resumida de la línea de comandos
`-n`	`--numeric`	Presenta la información numérica (por ejemplo, las direcciones IP), en lugar de intentar resolver a un *host*, puerto o nombre de usuario

Tabla 4.6 Opciones del comando **netstat**. (*continuación*)

Opción	Nemotecnia	Descripción
`-p`	`--program`	Muestra el nombre e identificador del proceso para cada *socket* de red enumerado
`-v`	`--verbose`	Imprime información adicional

Una característica útil de esta herramienta es la ayuda en línea. Sólo se debe invocar el comando con la opción `-h` o `help`:

```
netstat ;-help
```

El listado 4.2 presenta los resultados:

Listado 4.2 Resultados de **netstat --help.**

```
Usage: netstat [-veenNcCF] [<Af>] r  netstat {-V|--version|-h|--help}
       netstat [-vnNcaeol] [<Socket> ...]
       netstat { [-veenNac] -i | [ cnNe] -M|-s }

          -r, - route              display routing table
          -i,  -interfaces         display interface table
           g, --groups             display multicast group memberships
           s,  -statistics         display networking statistics
                                   (like SNMP)
           M, --masquerade         display masqueraded connections

           v, --verbose            be verbose
           n,  -numeric            dont resolve names
           N,  -symbolic           resolve hardware names
          -e, --extend             display other/more informations
           p,  --programs          display PID/Program name for sockets
           c, --continuous         continuous listing

           l, --listening          display listening server sockets
           a, --all, --listening   display all sockets (default:
                                   connected)
           o, --timers             display timers
           F, --fib                display Forwarding Infomation Base
(default)
           C, --cache              display routing cache instead of FIB

<Socket>={-t|- tcp} { u|--udp} { w|--raw} {-x}  unix} --ax25 --ipx
--netrom
<AF>=Use '-A <af>' or '--<af>' Default: inet
List of possible address families (which support routing):
  inet (DARPA Internet) ax25 (AMPR AX.25) netrom (AMPR NET/ROM)
```

Presentación de las sesiones de red activas

Uno de los servicios más importantes suministrados por **netstat** es la capacidad de ver las conexiones activas entre sistemas. Cualquier sesión TCP entre el *host* local y otro sistema puede seguuirse. Además, cualesquiera flujos de *sockets* (zócalos) que hayan sido creados se presentarán. Los flujos se usan como un canal de comunicación programa a programa. Para presentar las conexiones establecidas actualmente, se emite el **netstat** con la opción `-t`, como se muestra a continuación:

```
# netstat -t
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 110.orlando-11 12r:1052 192.215.123.37:www      ESTABLISHED
tcp        1      0 110.orlando-11 12r:1051 192.215.123.37:www      CLOSE
tcp        0      6 110.orlando-11-12r:1050 postoffice.worldn:pop-3 ESTABLISHED
tcp        0      0 110.orlando-11-12r:1049 www3.yahoo.com:www      ESTABLISHED
tcp        0      0 socrates.home.co:telnet durer.home.com:1033     ESTABLISHED
tcp        0      0 socrates.home.co:telnet durer.home.com:1032     ESTABLISHED
```

Recordemos, de la tabla 4.5, que la opción `-t` presentará la actividad del *socket* TCP. Como se indica, la salida del comando anterior incluye las conexiones en el sistema local. Cada conexión incluye información en cuanto a las direcciones locales y remotas, información estadística y estado de la conexión. Las direcciones locales y remotas se presentan para incluir el nombre del *host* y la información del puerto en el formato:

```
host:port
```

donde `nost` puede ser asignado como nombre de *host* desde `/etc/hosts` (o desde otro mecanismo de resolución de *host*, como NIS o DNS) o una dirección IP válida. `port` representa un puerto reservado, como se define en `/etc/services`, o un *socket* asignado por el sistema. La dirección local es la fuente y la dirección remota es el destino. Recuérdese que TCP usa cuatro elementos para crear una conexión y utiliza un modelo de máquina de estados como parte de todo el mecanismo de transporte de TCP. Como consecuencia, `monet:telnet` y `rembrandt:1036` se consideran una conexión. Desde el campo `State`, podemos ver que esta conexión se encuentra en estado `ESTABLISHED` (establecido), que significa que está pasando datos en ambos sentidos.

Recordemos que TCP usa una máquina de estados para controlar cada uno de los estados definidos, y que podemos usar el comando **netstat** para rastrear y presentar el estado de cada conexión

TCP. La tabla 4.7 muestra los estados más comunes e incluye una descripción general de cada uno.

Tabla 4.7 Estados TCP mostrados con **netstat.**

Estado	Descripción
ESTABLISHED	La conexión es operacional
LISTEN	Un servicio o aplicación está esperando una conexión de un cliente
SYN_SENT	Un sistema local quiere abrir una conexión remota
SYN_RCVD	Un sistema remoto quiere abrir una conexión
FIN_WAIT_1	Un sistema local está en proceso de cerrar una conexión
FIN_WAIT_2	Un sistema local está en proceso de cerrar una conexión
CLOSE_WAIT	Un sistema remoto quiere cerrar una conexión
LAST_ACK	Paso final a CLOSE_WAIT
TIMED_WAIT	Paso final a FIN_WAIT_1 o FIN_WAIT_2
UNKNOWN	El estado del *socket* es desconocido

El comando **netstat** anterior sólo presentó conexiones que están o estuvieron en estado ESTABLISHED. A veces es útil enumerar todos los servicios que están disponibles y activos en un sistema. Esto se puede lograr usando **netstat** con la opción -a, como se muestra más adelante.

Nótese que la siguiente salida se ha reducido para hacerla más legible. Ejecutar este comando en muchos sistemas producirá una lista mayor porque incluirá también las interfaces de flujo. Sin embargo, podemos usar las opciones -t y -u para refinar aún más la salida, a fin de incluir sólo los *sockets* TCP y UDP. La siguiente salida proporciona una lista de ambos servicios, UDP y

TCP, independientemente de sus estados de conexión. Esto es útil porque no siempre es obvio cuál protocolo de transporte usa un servicio en particular.

```
#netstat -a -t -u
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 socrates.home.co:telnet durer.home.com:1033 ESTABLISHED
tcp        0      0 110.orlando-11-1:domain *:*                   LISTEN
tcp        0      0 *:1048                  *:*                   LISTEN
tcp        0      0 *:1047                  *:*                   LISTEN
tcp        0      0 *:1046                  *:*                   LISTEN
tcp        0      0 *:1045                  *:*                   LISTEN
tcp        0      0 *:1044                  *:*                   LISTEN
tcp        0      0 *:1037                  *:*                   LISTEN
tcp        0    710 socrates.home.co:telnet durer.home.com:1032 ESTABLISHED
tcp        0      0 *:6000                  *:*                   LISTEN
tcp        0      0 *:nntp                  *:*                   LISTEN
tcp        0      0 *:www                   *:*                   LISTEN
tcp        0      0 *:smtp                  *:*                   LISTEN
tcp        0      0 *:713                   *:*                   LISTEN
tcp        0      0 *:1024                  *:*                   LISTEN
tcp        0      0 *:683                   *:*                   LISTEN
tcp        0      0 *:678                   *:*                   LISTEN
tcp        0      0 *:673                   *:*                   LISTEN
tcp        0      0 *:652                   *:*                   LISTEN
tcp        0      0 *:printer               *:*                   LISTEN
tcp        0      0 10.0.2.205:domain       *:*                   LISTEN
tcp        0      0 10.0.2.202:domain       *:*                   LISTEN
tcp        0      0 socrates.home.co:domain *:*                   LISTEN
tcp        0      0 localhost:domain        *:*                   LISTEN
tcp        0      0 *:linuxconf             *:*                   LISTEN
tcp        0      0 *:auth                  *:*                   LISTEN
tcp        0      0 *:finger                *:*                   LISTEN
tcp        0      0 *:login                 *:*                   LISTEN
tcp        0      0 *:shell                 *:*                   LISTEN
tcp        0      0 *:telnet                *:*                   LISTEN
tcp        0      0 *:ftp                   *:*                   LISTEN
tcp        0      0 *:sunrpc                *:*                   LISTEN
udp        0      0 110.orlando-11-1:domain *:*
udp        0      0 *:xdmcp                 *:*
udp        0      0 localhost:1119          *:*
udp        0      0 *:800                   *:*
udp        0      0 *:1022                  *:*
udp        0      0 *:714                   *:*
udp        0      0 *:1023                  *:*
udp        0      0 *:1026                  *:*
udp        0      0 *:2049                  *:*
udp        0      0 *:681                   *:*
udp        0      0 *:676                   *:*
udp        0      0 *:671                   *:*
udp        0      0 *:661                   *:*
```

(*continúa*)

```
udp        0      0 *:650                       *:*
udp        0      0 *:1024                      *:*
udp        0      0 10.0.2.205:domain           *:*
udp        0      0 10.0.2.202:domain           *:*
udp        0      0 socrates.home.co:domain     *:*
udp        0      0 localhost:domain            *:*
udp        0      0 *:snmp                      *:*
udp        0      0 *:ntalk                     *:*
udp        0      0 *:talk                      *:*
udp        0      0 *:sunrpc                    *:
```

Bajo el encabezado TCP, no sólo se presentan las dos conexiones TCP del ejemplo anterior, sino que también se incluyen los servicios adicionales. Cualesquiera servicios enumerados en el estado `LISTEN` están esperando conexiones entrantes que son generalmente conocidas como recursos basados en el servidor. Cuando un servicio está esperando solicitudes de la red, es libre de tener acceso a conexiones de cualquier dirección remota. Por eso `*.*` se enumera bajo el campo *Foreign Address*. Los servidores también colocan `*` por lo general en la porción de *host* local para indicar además que el servidor es libre de establecer una conexión si un cliente hizo una solicitud. Cuando una solicitud de un cliente se envía a un servidor, el servidor hace una copia de sí mismo para manejar la solicitud, y continúa escuchando las solicitudes adicionales de clientes. Así, cuando esto ocurre, **netstat** presenta múltiples ejemplos del mismo servicio, como se muestra a continuación:

```
netstat -a  | grep ftp
tcp        0      0 socrates.home.:ftp-data     durer.home.com:1034  TIME_WAIT
tcp        0      0 socrates.home.com:ftp       durer.home.com:1033  ESTABLISHED
tcp        0      0 *:ftp                       *:*    LISTEN
```

El comando anterior emite un **netstat** y canaliza la salida en el comando **grep**, el cual explora la entrada para la cadena `ftp`. Como resultado, se presentan todas las líneas con la cadena `ftp`. En la salida anterior, el servidor `FTP` todavía está escuchando las solicitudes entrantes de conexión mientras se establece una sesión `FTP` con `socrates`.

Bajo el encabezado UDP en el ejemplo de salida anterior, sólo se presentan una dirección local y un campo de estado; la dirección foránea (*Foreign Address*) no se especifica. Esto se debe a que UDP es un protocolo sin conexión y, por tanto, no requiere información de la dirección remota. Igualmente, nótese que ninguna información estadística está disponible para UDP. Ésta es

otra indicación de que UDP es fundamentalmente diferente en el diseño y no necesita este tipo de información.

A pesar de la enorme cantidad de información provista con la opción `a`, **netstat** puede usarse para proporcionar una rápida verificación, con el fin de asegurar que los servicios correctos están corriendo en un sistema dado. Explorando la salida de **netstat**, el administrador de red puede detectar fácilmente cualquier servicio que no debería estar corriendo. Por ejemplo, muchas organizaciones consideran la utilidad `finger` como un riesgo de seguridad porque puede proporcionar información de la cuenta de usuario a cualquiera que la pida. Una vez detectado con **netstat**, el servicio `finger` puede inhabilitarse modificando el archivo de configuración de red `/etc/inetd.conf`.

Para presentar los flujos definidos en el sistema, se emite el comando **netstat** con la opción `--unix`. La salida incluye las interfaces de flujo de *socket* UNIX. Como estas conexiones se usan principalmente para la comunicación interprocesos, su uso y función específicos no se describirán en detalle. Dado que el número de flujos usado en un sistema UNIX puede ser importante, la salida del comando **netstat** puede ser bastante grande. Como resultado, la siguiente salida muestra sólo algunas líneas frente a lo que podría presentarse comúnmente:

```
unix  1      [ ]    STREAM  CONNECTED  2399        /dev/log
unix  1      [ ]    STREAM  CONNECTED  2384        /tmp/.ICE-unix/963
unix  1      [ N ]  STREAM  CONNECTED  2364        /tmp/.X11-unix/X0
unix  1      [ ]    STREAM  CONNECTED  2220        /tmp/orbit-root/orb-
  11931020341330722701
unix  1      [ ]    STREAM  CONNECTED  2217        /tmp/orbit-root/orb
  2122911451756745208
unix  1      [ ]    STREAM  CONNECTED  2213        /tmp/orbit-root/orb
  16956010373298973
unix  1      [ ]    STREAM  CONNECTED  2206        /tmp/.X11-unix/X0
unix  1      [ ]    STREAM  CONNECTED  2202        /tmp/orbit-root/orb
  2122911451756745208
unix  1      [ ]    STREAM  CONNECTED  2196        /tmp/orbit-root/orb-
  16956010373298973
unix  1      [ ]    STREAM  CONNECTED  2197        /tmp/orbit-root/orb-
  13881122161642635705
unix  1      [ ]    STREAM  CONNECTED  2188        /tmp/.X11-unix/X0
unix  1      [ ]    STREAM  CONNECTED  2179        /tmp/orbit-root/orb
  16956010373298973
```

Estos flujos han sido creados por programas y aplicaciones del sistema como mecanismo para comunicarse entre sí y con otros programas.

Una característica muy útil de **netstat** es la opción `-p`, la cual muestra el nombre del proceso o programa asociado que tiene abierto un sistema o lo está usando. El comando

```
# netstat -t -p -a
```

produce esta salida:

```
(Not all processes could be identified, non-owned process info
will not be shown, you would have to be root to see it all.)
Active Internet connections (servers and established)
Proto Recv-Q Send Q  Local Address      Foreign Address            State
PID/Program name
tcp        0    285  socrates.home.co:telnet durer.home.com:1032 ESTABLISHED
  906/in.telnetd
tcp        0      0  *:1036                  *:*                LISTEN
  846/gnomepager_appl
tcp        0      0  *:1035                  *:*                LISTEN
  843/gen_util_applet
tcp        0      0  *:1034                  *:*                LISTEN 821/gmc
tcp        0      0  *:1033                  *:*                LISTEN 823/
  gnome-name-serv
tcp        0      0  *:1032                  *:*                LISTEN 812/
  panel
tcp        0      0  *:1025                  *:*                LISTEN 766/
  gnome-session
tcp        0      0  *:6000                  *:*                LISTEN 738/X
tcp        0      0  *:nntp                  *:*                LISTEN 685/
  innd
tcp        0      0  *:www                   *:*                LISTEN 602/
  httpd
tcp        0      0  *:smtp                  *:*                LISTEN
  573/sendmail: accep
tcp        0      0  *:713                   *:*                LISTEN 536/amd
tcp        0      0  *:1024                  *:*                LISTEN -
tcp        0      0  *:683                   *:*                LISTEN
  495/rpc.mountd
tcp        0      0  *:678                   *:*                LISTEN
  495/rpc.mountd
tcp        0      0  *:673                   *:*                LISTEN
  495/rpc.mountd
tcp        0      0  *:652                   *:*                LISTEN
  473/rpc.statd
tcp        0      0  *:printer               *:*                LISTEN 455/lpd
tcp        0      0  10.0.2.205:domain       *:*                LISTEN 441/
  named
tcp        0      0  10.0.2.202:domain       *:*                LISTEN 441/
  named
tcp        0      0  socrates.home.co:domain *:*                LISTEN 441/
  named
tcp        0      0  localhost:domain        *:*                LISTEN 441/
  named
tcp        0      0  *:linuxconf             *:*                LISTEN 397/
  inetd
tcp        0      0  *:auth                  *:*                LISTEN 397/
  inetd
```

(continúa)

```
tcp        0      0 *:finger                *:*                     LISTEN 397/
  inetd
tcp        0      0 *:login                 *:*                     LISTEN 397/
  inetd
tcp        0      0 *:shell                 *:*                     LISTEN 397/
  inetd
tcp        0      0 *:telnet                *:*                     LISTEN 397/
  inetd
tcp        0      0 *:ftp                   *:*                     LISTEN 397/
  inetd
tcp        0      0 *:sunrpc                *:*                     LISTEN 290/
  portmap
```

Una vez ejecutada, se agregan columnas adicionales a la salida normal de **netstat**. Ellas incluyen los campos PID (identificación del proceso) y `Program name` (nombre del programa). Como se puede observar, ahora es más fácil rastrear *sockets* y encontrar cúal proceso y/o programa los está usando. Nótese que esta versión de **netstat** le advierte al usuario que no todos los procesos pueden enumerarse, a menos que el usuario esté corriendo **netstat** como `root`. La salida anterior se obtuvo a partir de la ejecución del comando como usuario root, pero el mensaje se presenta de todas formas. Desde un punto de vista práctico, este mensaje debería ser ignorado.

Presentación de información de interfaz

El comando `netstat` puede obtener igualmente detalles en la configuración de la interfaz de red y conteos rudimentarios del paquete. La opción de línea de comandos `-i` obtiene una lista de cada interfaz definida en el sistema, una interfaz por línea:

```
#netstat -i
Kernel Interface table
Iface MTU  Met  RX-OK  RX-ERR  RX-DRP  RX-OVR  TX-OK  TX-ERR  TX-DRP  TX-OVR  Flg
eth0  1500   0      0       0       0       0      1       0       0  BRU
eth0: 1500   0      - no statistics available -                              BRU
eth1  1500   0   3946       0       0       0    138       0       0       0  BRU
lo    3924   0    192       0       0       0    192       0       0       0  LRU
```

Como se puede observar, **netstat** presenta parte de la misma información que proporciona el comando **ifconfig**, más algunas estadísticas básicas en cuanto a características de operación de cada interfaz, específicamente, el nombre de la interfaz, la unidad máxima de transferencia (MTU), la dirección de red o de destino, y la dirección de la interfaz. Además, presenta un conteo

del número total de paquetes de entrada, paquetes de error de entrada, paquetes de entrada abandonados, y el conteo de entrada de sobreflujo. Contiene los mismos contadores para los paquetes transmitidos. El campo `Flg` contiene un listado condensado de las opciones de configuración de interfaz, tal y como fueron habilitadas y reportadas por el comando **ifconfig**.

Los campos `RX-OK` (recibido) y `TX OK` (transmitido) representan la recepción y transmisión de tráfico válido a lo largo de la interfaz, respectivamente. Los campos `RX ERR` y `TX-ERR` indican cualesquiera paquetes de error de entrada y salida que hayan ocurrido en la interfaz; esto incluye, por ejemplo, cualesquiera paquetes pequeños (aquellos que son más pequeños que el tamaño estándar) y otros errores. Los campos `RX DRP` y `Tx DRP` son contadores que representan problemas con la transmisión de paquetes en la interfaz. En la salida anterior, nótese que la interfaz `eth0` reporta un número de errores de paquete de salida. En este caso, estos errores se están generando debido a que la interfaz no está conectada físicamente a una red, aunque el sistema esté intentando enviar paquetes. Algunos sistemas UNIX no pueden detectar cuándo una interfaz está realmente conectada a una red. Ésta también es la razón por la que los campos `RX OK` y `TX OK` son cero, lo cual indica que no se ha enviado ni recibido tráfico a lo largo de esta interfaz.

El campo `TX ERR` indica el número de colisiones (u otros errores de transmisión) que hayan ocurrido, tal como lo ha registrado el sistema. Una colisión ocurre cuando dos o más dispositivos intentan transmitir paquetes casi al mismo tiempo. Después de que esto sucede, se envía una señal de congestión para informarles a todos los dispositivos en la red que ha ocurrido una colisión y que cualquier transmisión debe ser detenida de manera breve; y luego, después de determinar al azar intervalos de tiempo, se intentará nuevamente. Esto se conoce como *back-off* y es el mecanismo que emplean los dispositivos para reanudar las operaciones normales. Las colisiones sólo ocurren con las tecnologías de red de difusión, como Ethernet. Cuando el campo `TX ERR` no es cero, indica que la interfaz ha registrado colisiones en las que estaba directamente involucrada. Es importante notar que este campo no representa todas las colisiones que hayan ocurrido en la red, ya que el sistema no puede contar siempre el número de mensajes de congestión transmitidos como resultado de una colisión causada por otros sistemas.

Los campos `RX DRP` y `TX DRP` representan paquetes que fueron descartados antes de ser recibidos o transmitidos. Estos campos

son útiles en situaciones en las que el sistema está realizando funciones de enrutamiento, donde los paquetes perdidos o descartados podrían causar problemas de conectividad entre sistemas o redes. Otro ejemplo en que puede ser importante hacer seguimiento a estos contadores se presenta cuando el sistema es un servidor, donde el tráfico de red puede ser significativo. En la práctica, los campos no son tan importantes para un sistema que pueda usarse como estación de trabajo de usuario único. Los campos `RX OVR` y `TX-OVR` proporcionan contadores para paquetes que causaron condiciones de sobreflujo para el software de trabajo en red. De nuevo, estas condiciones son críticas sólo cuando el sistema al que se le está haciendo seguimiento se considera crítico.

Cuando se definen interfaces lógicas (o seudointerfaces) en el sistema, **netstat** lista o enumera cada interfaz como una entrada por separado. Sin embargo, podrá notarse que, según el ejemplo anterior, **netstat** no recolecta la información estadística para estas interfaces. Como resultado, aparece en pantalla el mensaje `no statistics available` (estadísticas no disponibles). En todos los otros casos, **netstat** muestra interfaces lógicas con la misma información de las interfaces normales. Esto incluye, por ejemplo, los códigos de campo de la interfaz (`Flg`).

Algunos de los totales de paquetes presentados desde **netstat** casi coinciden con los valores obtenidos directamente del agente Linux SNMP conocido como `snmpd`. El programa `snmpd` es un agente que soporta objetos MIB que corresponden a objetos reportados por **netstat**. Los objetos MIB no pueden transformarse a los campos del contador de paquete **netstat** y se enumeran en la tabla 4.8. *Véase* el capítulo 7, "Uso de los agentes SNMP", para más detalles en cuanto a `snmpd` y los objetos MIB enumerados.

Tabla 4.8 Objetos MIB del agente Linux que corresponden a la salida **netstat**.

Campo netstat	Objetos MIB
RX-OK	IfInUcastPkts + ifInNUcastPkts
RX-ERR	IfInErrors
RX-DRP	IfInDiscards
TX-OK	IfOutUcastPkts + ifOutNUcastPkts
TX-ERR	IfOutErrors
TX-DRP	IfOutDiscards

Si el sistema tiene una gran cantidad de interfaces y sólo se tiene interés en una en particular, se debe usar la opción de línea de comandos `-interfaces` para especificar la que se quiere. Nótese que el manual en línea describe específicamente esta característica, pero por alguna razón **netstat** ignora cualesquiera opciones del comando interfaz.

Presentación de la información de enrutamiento

El sistema utiliza la tabla de enrutamiento para determinar la ruta que se usará para enviar paquetes IP a host o redes en particular. Normalmente, los sistemas estan configurados con un enrutador predeterminado para que las decisiones de enrutamiento sean sencillas y simples. Sin embargo, hay algunos casos en los que una máquina tiene más de una interfaz, y cada una está conectada a una red IP diferente. En esta situación, el sistema podría estar reenviando paquetes IP (enrutando) entre estas redes, lo cual hace que, la función de enrutamiento se vuelva un poco más compleja. Como parte del sistema de enrutamiento, se define una tabla de enrutamiento que puede presentarse a medida que surja la necesidad. Una de las formas principales para examinar esta tabla es con la opción `-r`:

```
# netstat -r
Kernel IP routing table
Destination     Gateway         Genmask         Flags  MSS  Window  irtt  Iface
199.70.195.41   *               255.255.255.255 UH       0       0     0  ppp0
10.0.2.201      *               255.255.255.255 UH       0       0     0  eth0
10.0.2.0        *               255.255.255.0   U        0       0     0  eth0
127.0.0.0       *               255.0.0.0       U        0       0     0  lo
default         199.70.195.41   0.0.0.0         UG       0       0     0  ppp0
```

La salida anterior provenía de un sistema servidor que contiene dos interfaces de red separadas. En este ejemplo, la tabla de enrutamiento incluye una red de destino, puerta de enlace (o enrutador), máscara de red, algunas banderas de estado, dos campos de tamaño, un valor métrico y la interfaz con la que la ruta está asociada. El campo `destination` (destino) especifica la red para la cual se establece la ruta. El campo `gateway` (puerta de enlace) muestra la dirección IP o nombre del *host* del enrutador que reenvía paquetes a la dirección IP listada en la columna `Destination`. Un `*` indica que el enrutador no ha sido configurado para la red asociada. Si se muestra una dirección IP o nombre de *host* en este campo, entonces, se ha configurado un enrutador.

El campo `Genmask` muestra la máscara de red que se ha configurado para esta interfaz. Esta máscara se utiliza como una máscara de subred para calcular la dirección de red especificada en la columna `Destination`.

El campo `Flags` (banderas) muestra la información de estado en cuanto a la ruta. La bandera `U` indica que la ruta está arriba y activa. La bandera `H` muestra que la entrada de ruta se refiere a un sistema *host*, no a un enrutador real. Con UNIX, hay siempre una ruta hacia el sistema local, la cual es usada internamente por el software de trabajo en red. La bandera `G` indica que la ruta es a través de una puerta de enlace o enrutador externo.

Nota Los términos `route` (ruta) y `gateway` (puerta de enlace) se emplean de manera intercambiable.

Cuando listamos las tablas de enrutamiento desde una estación de trabajo que contiene una sola interfaz, podemos ver las siguientes entradas:

```
# netstat -r
Kernel IP routing table
Destination     Gateway         Genmask         Flags  Metric  Ref Use  Iface
199.70.195.41   *               255.255.255.255 UH     0       0        0 ppp0
10.0.2.201      *               255.255.255.255 UH     0       0        0 eth0
10.0.2.0        *               255.255.255.0   U      0       0        0 eth0
127.0.0.0       *               255.0.0.0       L      0       0        0 lo
default         199.70.195.41   0.0.0.0         UG     0       0        0 ppp0
```

En este caso, se ha establecido una ruta predeterminada en `199.70.195.41`, que resulta ser una conexión con un ISP local, usando el protocolo punto a punto, PPP. Cuando un sistema contiene una sola interfaz, puede emplearse una ruta predeterminada como un método abreviado para especificar la única ruta de salida de la red local. Sin la entrada predeterminada, cada red a la que el sistema debe conectarse requerirá una entrada de enrutamiento separada. Cuando se activa el enlace `ppp0` (ya sea manual o automáticamente), el software PPP instala automáticamente la ruta predeterminada.

El campo `MMS` representa el tamaño máximo de segmento (*Maximum Segment Size*, MSS) para una sesión o conexión TCP. Normalmente, con **netstat**, este campo contiene un valor cero. El campo `Window` (ventana) controla el tamaño de la ventana TCP para una conexión que usa esta ruta; en general, esto sirve para

ciertos protocolos WAN u otros controladores de red que tengan problemas en la manipulación de marcos consecutivos (*back-to-back*). De nuevo, este campo normalmente tiene un valor de cero.

El campo `irtt` muestra el tiempo de ida y vuelta inicial (*initial round-trip time*, IRTT) para una sesión o conexión TCP; también se utiliza para protocolos de red WAN. El comando **netstat** muestra el valor cero. El último campo (`Iface`) muestra la interfaz de red a la que pertenece la ruta. Es importante notar que dentro de las tablas de enrutamiento, muchas rutas podrían usar la misma interfaz. De hecho, el ejemplo anterior no muestra menos de tres rutas que utilizan la misma interfaz. Esto es normal y apropiado, ya que la función de enrutamiento está involucrada con el reenvío de paquetes IP desde una red a otra, independientemente de la red física que se emplee o la ruta que se recorra. Esto, aunque de forma muy pequeña, ilustra la modularidad de los protocolos TCP/IP y el software de trabajo en red.

Además, el comando **netstat** soporta un formato alterno cuando se listan las tablas de enrutamiento. El comando **route** UNIX altera las tablas de enrutamiento del sistema y también puede presentar información de enrutamiento. El comando **netstat** puede presentar información de enrutamiento en el mismo formato del comando **route**. Por tanto, el comando **netstat -r**, cuando se combina con la opción de línea de comandos `-e`, y se ejecuta en el mismo sistema de donde se obtuvo la salida anterior, presenta las tablas de enrutamiento del sistema.

El comando: **netstat -e -r** presenta lo siguiente:

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags  Metric Ref    Use Iface
199.70.195.41   *               255.255.255.255 UH     0      0        0 ppp0
10.0.2.201      *               255.255.255.255 UH     0      0        0 eth0
10.0.2.0        *               255.255.255.0   U      0      0        0 eth0
127.0.0.0       *               255.0.0.0       U      0      0        0 lo
default         199.70.195.41   0.0.0.0         UG     0      0        0 ppp0
```

El campo `Metric` registra la distancia o cuenta de saltos a la red o *host* objetivo. `Ref` es el número total de rutas que comparten la misma dirección de enlace de datos o física, y el campo `Use` muestra el número de paquetes transmitidos desde que fue activada la interfaz. Desafortunadamente, el comando **netstat** no parece tener la capacidad de obtener ninguno de los valores válidos para

los campos `Ref` y `Use`, ya que ellos siempre son establecidos como cero.

Presentación de la información de multidifusión

La multidifusión es un mecanismo que soporta de una manera muy eficiente la entrega de tráfico de gran volumen a una red y estaciones de trabajo asociadas. Un grupo de multidifusión es una colección definida de estaciones de trabajo y enrutadores de multidifusión que reenvían tráfico mediante una dirección IP de multidifusión especial. La opción `-g` presenta información de enrutamiento de multidifusión relacionada con los grupos de enrutamiento y las interfaces que se han definido en el sistema. Si se utiliza esta opción, el comando **netstat -g** mostrará los grupos de multidifusión configurados actualmente:

```
IPv4 Group  Memberships
Interface   RefCnt      Group
---         --          ---
lo          1           224.0.0.1
eth0        1           224.0.0.1
eth1        1           224.0.0.1
```

En este ejemplo, cada una de las interfaces definidas en el sistema es un miembro del grupo de multidifusión predeterminado, conocido como `224.0.0.1` o `ALL-SYSTEMS.MCAST.NET`. Este grupo, que es un grupo de multidifusión estándar, se utiliza para enviar tráfico de multidifusión a todos los sistemas en una red local. Así, cuando una aplicación use la dirección `224.0.0.1` para transmitir tráfico, este sistema recibirá la información. En ocasiones en que se emplea multidifusión usando aplicaciones de multidifusión estándar, pueden definirse los grupos de multidifusión adicionales para restringir el tráfico de multidifusión sólo para aquellos sistemas en los que se requiere la información.

Presentación de las estadísticas del protocolo

El comando **netstat** puede usarse para presentar estadísticas del protocolo. La opción `statistics`, por sí misma, presentará los protocolos soportados, incluidos TCP, UDP y RAW. RAW es una combinación de los paquetes IP e ICMP y pueden presentarse por separado usando la palabra clave **raw.**

```
# netstat --statistics
Ip:
      3003 total packets received
      0 forwarded
      0 incoming packets discarded
      212 incoming packets delivered
      2847 requests sent out
Icmp:
      489 ICMP messages received
      0 input ICMP message failed.
      ICMP input histogram:
          destination unreachable: 486
          echo replies: 3
      487 ICMP messages sent
      0 ICMP messages failed
      ICMP output histogram:
          destination unreachable: 487
Tcp:
      0 active connections openings
      0 passive connection openings
      0 failed connection attempts
      0 connection resets received
      1 connections established
      2295 segments received
      1700 segments send out
      2 segments retransmited
      0 bad segments received.
      0 resets sent
Udp:
      171 packets received
      2 packets to unknown port received.
      0 packet receive errors
      657 packets sent:
```

Por otro lado, podría presentarse tan sólo uno de los protocolos listados, en lugar de la lista completa. En esta situación, se usa una de las palabras clave opcionales **--raw**, **--tcp** o **--udp.** Después de emplear una de estas palabras clave, será presentado sólo ese protocolo. Por ejemplo, para presentar sólo paquetes IP e ICMP, se usa el siguiente comando:

```
netstat --statistics --raw
```

Después de utilizar este comando, se presentará en pantalla lo siguiente:

```
Ip:
      3360 total packets received
      0 forwarded
```

(*continúa*)

```
        0 incoming packets discarded
        213 incoming packets delivered
        3090 requests sent out
    Icmp:
        489 ICMP messages received
        0 input ICMP message failed.
        ICMP input histogram:
            destination unreachable: 486
            echo replies: 3
        487 ICMP messages sent
        0 ICMP messages failed
        ICMP output histogram:
            destination unreachable: 487
```

Opciones adicionales de línea de comandos

Con el comando **netstat** están disponibles algunas opciones adicionales; éstas incluyen la opción `-t`, la cual se usa para controlar la presentación de las direcciones numéricas en lugar de tratar de determinar el nombre simbólico del *host*, puertos o los nombres de usuarios. Esta opción se utiliza cuando los servicios de resolución del nombre, como DNS o NIS, no funcionan correctamente o no están disponibles. El comportamiento predeterminado para **netstat** es presentar el nombre del *host* y otra información simbólica. El comando **netstat** presentará las conexiones TCP actuales en el sistema local:

```
Proto  Recv-Q Send-Q  Local Address           Foreign Address         State
tcp         0    126  socrates.home.co:telnet durer.home.com:1035     ESTABLISHED
```

Sin embargo, cuando se agrega la opción `-t`, aparece en pantalla lo siguiente:

```
Proto Recv-Q Send-QLocal Address       Foreign Address         State
tcp         0    126 10.0.2.201:23     10.0.2.10:1035          ESTABLISHED
```

Nótese igualmente que no sólo se presentó la información de la dirección IP, sino que también se presentaron los números de puertos. Normalmente, **netstat** examinará el archivo `/etc/services` para determinar las transformaciones entre los nombres y números de puertos, y mostrará los nombres de los puertos. La opción `-t` suprimió estas transformaciones para puertos. Otra opción útil es la opción `-c`, la cual alterna continuamente presentaciones de información. Si queremos hacer seguimiento constante a la

información presentada en el ejemplo anterior, usamos el siguiente comando:

```
# netstat -c -t
```

para generar la siguiente salida:

```
Active Internet connections (w/o servers)
Proto  Recv-Q Send Q  Local Address           Foreign Address         State
tcp         0    126 socrates.home.co:telnet durer.home.com:1035     ESTABLISHED

Active Internet connections (w/o servers)
Proto  Recv-Q Send Q  Local Address           Foreign Address         State
tcp         0    124 socrates.home.co:telnet durer.home.com:1035     ESTABLISHED

Active Internet connections (w/o servers)
Proto  Recv-Q Send-Q  Local Address           Foreign Address         State
tcp         0      0 socrates.home.com:ftp   durer.home.com:1036     ESTABLISHED
tcp         0    126 socrates.home.co:telnet durer.home.com:1035     ESTABLISHED

Active Internet connections (w/o servers)
Proto  Recv-Q Send-Q  Local Address           Foreign Address         State
tcp         0      0 socrates.home.com:ftp   durer.home.com:1036     ESTABLISHED
tcp         0    124 socrates.home.co:telnet durer.home.com:1035     ESTABLISHED

Active Internet connections (w/o servers)
Proto  Recv-Q Send-Q  Local Address           Foreign Address         State
tcp         0      0 socrates.home.:ftp-data durer.home.com:1037     ESTABLISHED
tcp         0      0 socrates.home.com:ftp   durer.home.com:1036     ESTABLISHED
tcp         0    124 socrates.home.co:telnet durer.home.com:1035     ESTABLISHED
```

Desde esta salida podemos ver que primero se establece una única conexión `telnet` entre `socrates` y `durer`. Luego arranca una sesión `FTP` entre estos sistemas, como se muestra a continuación:

```
tcp         0      0 socrates.home.com:ftp durer.home.com:1036      ESTABLISHED
```

Una vez establecida la sesión `ftp`, el usuario ejecutó y/o transfirió un archivo. Esto puede verse con la conexión de *ftp-data*, usando el puerto 1037, el cual se crea cuando un usuario emite un comando **ftp** o transfiere un archivo. Para detener la presentación continua de la salida, se usa la secuencia **^c** (**control+c**). Como se puede imaginar, ésta es una forma muy útil de hacer seguimiento a la conectividad de la red.

Herramienta ping

Descripción general

El comando **ping** suministra dos servicios básicos. Primero, se puede usar para determinar si un nivel básico de conectividad

está disponible entre uno o más puntos finales o sistemas. La herramienta *ping* puede usarse para determinar si un dispositivo remoto se puede alcanzar en una red desde el sistema local, y ayuda a depurar problemas de conectividad entre sistemas. Segundo, puede proporcionar estadísticas de desempeño de red rudimentarias, las cuales pueden usarse para diagnosticar problemas de red relacionados con el tráfico. El término *ping* se deriva de la frase *packet internet groper* (programa de diagnóstico de Internet). La herramienta *ping* puede usarse en una de dos formas: especificando un nombre de *host* o dirección IP válido, o usando las opciones de línea de comandos con un nombre de *host* o dirección IP. Si se usa la primera forma, *ping* suministra una forma útil para determinar que un dispositivo remoto está disponible en la red.

Recuérdese que en el capítulo 3, "Conjunto de protocolos TCP/IP", mencionamos que *ping* usa el protocolo de mensajes de control de Internet (ICMP) para emitir solicitudes ICMP, y espera respuestas ICMP válidas. Debido a que ICMP es un protocolo requerido en la familia TCP/IP, *ping* puede usarse generalmente con todos los dispositivos que soportan TCP/IP y está disponible en muchos sistemas operativos y otros dispositivos de trabajo en red. Por ejemplo, un enrutador Cisco proporciona la capacidad de usar *ping* en otros dispositivos en la red. La herramienta *ping* es sólo una aplicación del lado del cliente; no se necesita ni se requiere software adicional para que funcione e interactúe directamente con la capa de protocolo del sistema remoto para cumplir su tarea.

Determinación de la disponibilidad del sistema

La herramienta *ping* puede usarse para determinar la disponibilidad general de cualquier dispositivo TCP/IP, aun si no tiene específicamente un sistema operativo general. Por ejemplo, para determinar si el *host* `rembrandt` se puede alcanzar, se emite el siguiente comando **ping:**

```
#ping durer
PING durer.home.com (10.0.2.10): 56 data bytes
64 bytes from 10.0.2.10: icmp_seq=0 ttl=128 time=0.9 ms
64 bytes from 10.0.2.10: icmp_seq=1 ttl=128 time=0.8 ms
64 bytes from 10.0.2.10: icmp_seq=2 ttl 128 time=0.8 ms
64 bytes from 10.0.2.10: icmp_seq=3 ttl=128 time=0.8 ms

 - durer.home.com ping statistics -
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 0.8/0.8/0.9 ms
```

En este caso, **ping** no presenta paquetes perdidos (*packet loss*) a `durer`, que resulta ser una impresora láser PostScript. Esto establece básicamente que `durer` está funcionando y operando de manera normal desde una perspectiva IP. El comportamiento predeterminado de **ping** significa que el usuario debe presionar **^c** (**control+c**) para detener la salida. Este mensaje significa generalmente que el software TCP/IP es operacional. Aunque *funcionando* indica que el sistema está visible en la red, no garantiza que estén disponibles otros servicios de red, como FTP o Telnet. Ésta es una distinción importante. La herramienta *ping* puede usarse sólo para determinar la conectividad básica del protocolo, no la disponibilidad de las aplicaciones o los servicios de alto nivel. De hecho, algunos sistemas responderán al **ping** aun antes de ser totalmente iniciados. Debe tenerse en cuenta que ninguna pieza de software puede determinar que todas las aplicaciones o servicios TCP/IP están instalados y operando en un sistema.

Si el *host* `durer` no se puede alcanzar, **ping** presentará el siguiente mensaje después de digitar **^c**:

```
PING rubens.home.com (10.0.2.220): 56 data bytes
 - rubens.home.com ping statistics  -
2 packets transmitted, 0 packets received, 100% packet loss
```

Normalmente, **ping** emite solicitudes ICMP por siempre; si no se recibe ninguna respuesta, genera el mensaje que se mostró antes sólo después de que el usuario interrumpe el comando. Esto es de alguna forma infortunado, ya que otras versiones de **ping** serán extemporáneas eventualmente, sin que el usuario tenga que interrumpir el comando de manera manual. Por fortuna, puede especificarse un número máximo o conteo del número total de solicitudes, lo que tiene el efecto de controlar **ping**, para que el usuario no tenga que intervenir manualmente. Si se utiliza la opción de línea de comandos `-c` con un argumento de `1`, el comando **ping** emitirá una sola solicitud a `rubens`:

```
# ping -c 1 rubens
```

y generará la siguiente salida, si este *host* está abajo:

```
PING rubens.home.com (10.0.2.220): 56 data bytes
   rubens.home.com ping statistics --
1 packets transmitted, 0 packets received, 100% packet loss
```

Esto es útil para que el usuario pueda determinar rápidamente el alcance de un *host* o anfitrión, sin perder tiempo adicional o ancho de banda de la red. Reducir el número de solicitudes **ping** es en general bueno para la red. A veces es preferible especificar el conteo de esta forma cuando se está usando **ping** en una secuencia de comandos shell donde emitir un **^c** sería difícil o inconveniente. Usar la opción de conteo es una forma ideal de obtener un muy buen promedio de la demora del viaje de ida y vuelta y determinar el rendimiento a través del tiempo.

Es interesante notar que si el *host* `rubens` no estuviera en la misma subred en la que el *host* emite el **ping**, es posible que el *host* esté funcionando correctamente, pero un dispositivo intermedio, como un enrutador de red, sea el responsable de la falta de conectividad. Yo llamo a este problema *connectivity fussiness* (minuciosidad de la conectividad). En este caso, **ping** no puede determinar por qué `rubens` no se puede alcanzar. Para entender este problema más a fondo, consideremos la red de ejemplo en la figura 4.1. Este diagrama de red muestra varios dispositivos conectados a dos redes diferentes que están interconectados a través del `enrutador A`. Cuando se emite una solicitud **ping** desde el `nodo B` en la `red A` para el `nodo D` en la `red B`, la solicitud pasa a través del `enrutador A`. Si el `enrutador A` debe dejar de funcionar, las solicitudes nunca alcanzarán al `nodo D`. Como consecuencia, el `nodo D` se vuelve inalcanzable desde la perspectiva del `nodo B`.

Debido a que **ping** puede verificar el alcance de cualquier dispositivo TCP/IP, podemos ahora emitir un **ping** para el `enrutador A`, con el fin de diagnosticar mejor el problema. Sondeando el enrutador más cercano al `nodo D`, aprenderemos que la pérdida de la conectividad es producida más posiblemente por la interfaz de red del `enrutador A` con la `red B`, y no por el mismo `nodo D`. Además, si producimos un **ping** a otros dispositivos en la `red B`, esto confirmaría que todos los dispositivos están fuera de alcance, y nos lleva a concluir que hay un problema con el `enrutador A`. Este ejemplo demuestra que los problemas de red pueden ser causados por otros dispositivos, en lugar de aquellos identificados fácilmente como remitente y receptor. Herramientas como **ping** ayudan a aislar las fuentes de enrutamiento y muchas otras fallas o problemas de red.

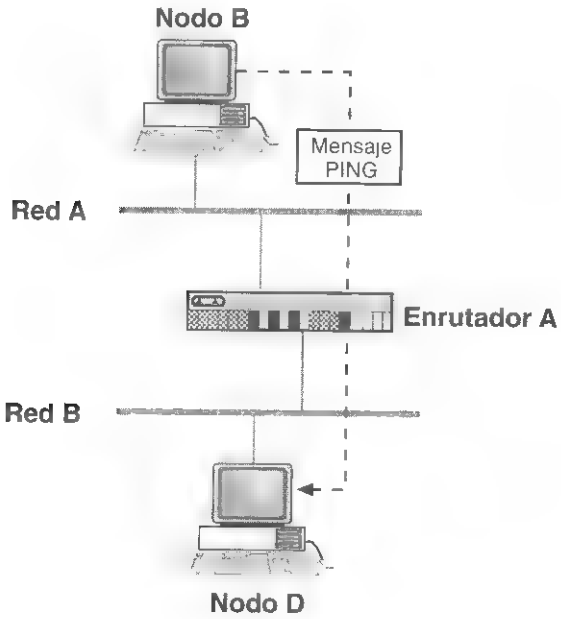


Figura 4.1 Problema de nodo fuera de alcance.

La segunda forma del comando **ping** proporciona varias opciones para controlar la funcionalidad adicional. La tabla 4.9 suministra una lista de las opciones de línea de comandos más populares disponibles.

Tabla 4.9 Opciones de comando **ping**.

Opción	Descripción
`-c`	Envía sólo cierto número de paquetes
`f`	Inunda la red con paquetes
`i`	Aumenta la cantidad de segundos entre una y otra solicitud
`-n`	Muestra las direcciones de red
`-p`	Especifica hasta 16 bytes para rellenar un paquete con datos personalizados

Tabla 4.9 Opciones de comando **ping**. (*continuación*)

Opción	Descripción
`-q`	Modo silencioso
`-R`	Alterna la opción de registro de ruta
`-s`	Especifica el número de bytes que van a ser enviados con cada solicitud

Determinación del desempeño de la red

El comando **ping** puede usarse para medir la cantidad de tiempo requerida para transmitir un mensaje a un destino remoto y la cantidad de tiempo requerida para obtener una respuesta. Este uso del comando **ping** mide esencialmente el rendimiento relativo de la ruta entre los dos dispositivos en un momento determinado. Éste no suministra, de ningún modo, un análisis detallado de los dispositivos o conectividad entre ellos. En su lugar, proporciona una ojeada a la condición general de la ruta en el punto donde se mide. Se podría decir que el desempeño de la red es como el mercado de valores. Un día está arriba y al siguiente abajo. La principal diferencia respecto a la volatilidad es si estamos hablando en términos de días o milisegundos. Una gran cantidad de factores pueden hacer que varíe el rendimiento de la red. Éstos incluyen usuarios demasiado agresivos en el uso de los recursos de la red, problemas del hardware, problemas de configuración del software, y otros relacionados.

El comando **ping** proporciona también medios de determinar los tiempos de respuesta del sistema, pero le toma un poco más de trabajo determinar si el problema de rendimiento observado está relacionado con un sistema lento específico o con una demora en algún otro componente de red. La herramienta *ping* muestra la salida que puede usarse para medir y reportar el tiempo de ida y vuelta y proporciona estadísticas de pérdida de paquete. Por defecto, **ping** emite una solicitud ICMP cada segundo al destino suministrado en la línea de comandos y reporta el estado de cada respuesta ICMP. La salida en el ejemplo incluye lo siguiente:

```
# ping didymus
PING didymus-gw2.home.com (10.0.2.127): 56 data bytes
```

(*continúa*)

```
64 bytes from 10.0.2.127: icmp_seq=0 ttl=255 time=1.2 ms
64 bytes from 10.0.2.127: icmp_seq=1 ttl=255 time=1.1 ms
64 bytes from 10.0.2.127: icmp_seq=2 ttl=255 time=1.2 ms
64 bytes from 10.0.2.127: icmp_seq=3 ttl=255 time=1.2 ms

    didymus gw2.home.com ping statistics --
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 1.1/1.1/1.2 ms
```

Este reporte informa el tamaño del paquete, el nombre del *host* o la dirección IP del dispositivo objetivo, un número de secuencia, un valor de tiempo de ida y vuelta, y un resumen estadístico. El valor del tiempo muestra el tiempo de ida y vuelta en milisegundos (milésimas de segundo) para cada respuesta recibida. La parte inferior del reporte calcula los tiempos de viaje mínimo, promedio y máximo para todas las respuestas, presentados igualmente en milisegundos. El total de la longitud del paquete ICMP transmitido a `didymus` es de 64 bytes. Éste es el tamaño predeterminado, el cual es generalmente suficiente. Sin embargo, puede ser necesario incrementar el tamaño del paquete para obtener una mejor medida de rendimiento. En este caso, puede especificarse un tamaño grande de paquete usando la opción línea de comandos `-s`. Por ejemplo, el comando:

```
# ping  s 100 didymus
```

emite las solicitudes ICMP con un tamaño de paquete de 100 bytes al *host* objetivo `didymus`. Esto podría ser necesario para obtener una mejor imagen del desempeño, ya que el rendimiento de la red puede diferir para los tamaños grandes de paquetes frente a valores menores. Cuando se ejecuta, este comando muestra lo siguiente:

```
PING didymus-gw2.home.com (10.0.2.127): 100 data bytes
108 bytes from 10.0.2.127: icmp_seq=0 ttl=255 time=2.7 ms
108 bytes from 10.0.2.127: icmp_seq=1 ttl=255 time=1.5 ms
108 bytes from 10.0.2.127: icmp_seq=2 ttl=255 time=1.3 ms
108 bytes from 10.0.2.127: icmp_seq=3 ttl=255 time=1.3 ms
108 bytes from 10.0.2.127: icmp_seq=4 ttl=255 time=1.3 ms
108 bytes from 10.0.2.127: icmp_seq=5 ttl=255 time=1.3 ms
108 bytes from 10.0.2.127: icmp_seq=6 ttl=255 time=1.3 ms
108 bytes from 10.0.2.127: icmp_seq=7 ttl=255 time=1.3 ms

    didymus-gw2.home.com ping statistics ---
8 packets transmitted, 8 packets received, 0% packet loss
round trip min/avg/max = 1.3/1.5/2.7 ms
```

Como se puede observar en esta salida, **ping** agrega ocho bytes de sobrecarga para cada paquete enviado; esto se determina restando los 100 bytes especificados con la opción `-s` de los 108 bytes transmitidos por **ping**. Nótese que los tiempos de respuesta no cambiaron mucho, a pesar de que usamos un tamaño de datos grande. Necesitaríamos incrementar el tamaño significativamente para observar una mayor demora en el procesamiento de los paquetes.

Nota

Si tal vez se haya notado que el nombre de *host* `tidymus` se usó en la línea de comandos, pero cuando **ping** resonó el nombre del *host*, mostró un nombre diferente, como `tidymus.gw2.home.com`. La razón de esto es que `tidymus` es un alias de `tidymus.gw2.home.com`, y usar el alias con muchos comandos UNIX da como resultado que el nombre oficial sea devuelto en su lugar. El alias se creó en el archivo `/etc/hosts`.

La herramienta *ping* usa un número de secuencia para mantener el rastro de las solicitudes y respuestas. A cada solicitud se le da el siguiente número en la secuencia y luego es cotejado con la respuesta correspondiente. Esta secuencia se usa para determinar la pérdida de paquetes si cualquier solicitud no recibe una respuesta apropiada. En general, la pérdida de paquete en una red pequeña sería muy rara y, si ocurre, podría indicar un problema relacionado con la red o con el sistema. Sin embargo, en una red o internet grande (internet con *i* minúscula) o en Internet, la pérdida de paquete es común y representa un estado normal de eventos. Dado un sitio popular de Internet, como se muestra a continuación, puede observarse cierta cantidad de pérdida de paquetes:

```
ping -c 10 www.whitehouse.gov
PING www.whitehouse.com (209.67.27.247): 56 data bytes
64 bytes from 209.67.27.247: icmp_seq=7 ttl=244 time=240.1 ms
64 bytes from 209.67.27.247: icmp_seq=8 ttl=244 time=240.1 ms
64 bytes from 209.67.27.247: icmp_seq=9 ttl=244 time=240.1 ms

    www.whitehouse.com ping statistics
10 packets transmitted, 3 packets received, 70% packet loss
round-trip min/avg/max = 240.1/240.1/240.1 ms
```

¡El reporte anterior indica que 70% de los paquetes enviados al sistema `www.whitehouse.gov` no tuvieron sus correspondientes respuestas! Se perdieron. En otras palabras, enviamos diez

paquetes, pero sólo se recibieron tres de vuelta; siete de diez es 70%. Una posible razón de esta pérdida visible de paquete es que algunos de los enrutadores críticos de Internet podrían estar muy ocupados o aun sobrecargados con el tráfico de la red. Como consecuencia, algunas de las solicitudes ICMP podrían ser descartadas, ya que las solicitudes expiraron antes de que fueran entregadas al destino final. Además, la carga relativa del dispositivo objetivo puede ser un factor, porque estos sistemas podrían no tener los recursos de cómputo para responder como es requerido a todas las solicitudes de red. Debido a la popularidad de este sitio, no es irrazonable pensar que los servidores y las redes que los conectan estuvieran todos muy ocupados o hasta sobrecargados. Una condición de sobrecarga ocurrirá cuando demasiados usuarios estén usando al mismo tiempo recursos del sistema.

A veces es deseable suministrar tiempo adicional para reconocer cada solicitud **ping**, en lugar de usar el valor predeterminado de un segundo. Si se desea tiempo adicional entre las solicitudes ICMP sucesivas, se puede usar la opción `-i`, seguida del valor deseado. El intervalo debe ser suficientemente largo para proporcionar la cantidad de tiempo requerida para que el sistema remoto responda. Cuando aumentamos el valor de extemporaneidad tal como se sugiere, generalmente notaremos menos pérdida de paquetes. El comando

```
ping -c 10 www.whitehouse.gov -i 5
```

adiciona cinco segundos de demora a cada solicitud, proporcionando así tiempo adicional para el procesamiento de las solicitudes a través de la red y al servidor de destino. Usando el comando anterior, se produce lo siguiente:

```
PING www.whitehouse.com (209.67.27.247): 56 data bytes
64 bytes from 209.67.27.247: icmp_seq 1 ttl=244 time=240.1 ms
64 bytes from 209.67.27.247: icmp_seq=2 ttl=244 time 240.1 ms
64 bytes from 209.67.27.247: icmp_seq=3 ttl=244 time=240.1 ms
64 bytes from 209.67.27.247: icmp_seq=4 ttl=244 time=240.0 ms
64 bytes from 209.67.27.247: icmp_seq=5 ttl=244 time=250.1 ms
64 bytes from 209.67.27.247: icmp_seq=6 ttl=244 time=240.1 ms
64 bytes from 209.67.27.247: icmp_seq=7 ttl=244 time=240.1 ms
64 bytes from 209.67.27.247: icmp_seq=8 ttl=244 time=240.2 ms
64 bytes from 209.67.27.247: icmp_seq=9 ttl=244 time=250.1 ms

--- www.whitehouse.com ping statistics --
10 packets transmitted, 9 packets received, 10% packet loss
round-trip min/avg/max = 240.0/242.3/250.1 ms
```

Como se observa en la salida, la pérdida de paquetes en este sitio fue reducida a 10%. Debe tenerse en cuenta que otros factores podrían también contribuir a la reducción, como usuarios que abandonan el sitio, o que la red no se esté usando. En general, aumentar la cantidad de tiempo para cada solicitud debería reducir la carga total en el sistema. Sin embargo, esto no garantiza que sea siempre el caso, ya que el sistema puede estar sobrecargado, hasta un punto en que no ayudaría realmente ninguna cantidad de tiempo adicional.

Opciones varias

Con la opción `-n`, **ping** presenta direcciones IP, en lugar de nombres de *hosts*. Esto es útil, por ejemplo, cuando problemas de red que involucran DNS impactan el uso de **ping**. Esta opción instruye a **ping** a no invocar la resolución del nombre de *host*, lo que permite que la herramienta funcione mientras el servicio de nombre está lento o temporalmente inhabilitado.

La opción `-R` habilita la opción de registro de ruta con el protocolo IP. Alternar el registro de ruta le informa a cada enrutador a lo largo de una ruta que coloque sus direcciones IP en el encabezado IP. Como consecuencia, se puede obtener una lista de enrutadores que se utilizaron para alcanzar el destino final. Éste es el mecanismo principal que utiliza el comando **traceroute**. Esta herramienta se describe más adelante en este capítulo. Otra opción interesante es el modo de inundación mediante el uso de la opción `-f`. Esta opción le dice a **ping** que intente inundar la red con solicitudes ICMP aproximadamente 100 veces por segundo o tan rápidamente como el destino remoto pueda procesar cada solicitud.

A propósito, tenga en cuenta este aviso de precaución; la opción `-f` puede ser algo peligrosa. Puede consumir una cantidad considerable de ancho de banda de red y causar que los sistemas desaparezcan de la red, ya que están muy cargados para responder a otras solicitudes de red. No se recomienda el uso de esta opción en una red en funcionamiento, cuando la pérdida de conectividad podría afectar las operaciones de negocios de usuarios individuales de la red. Además, no es razonable inundar otras redes con las que no se esté vinculado.

Teniendo en cuenta lo anterior, el comando

```
ping -c 100 -f durer
```

presenta la siguiente salida:

```
PING durer.home.com (10.0.2.10): 56 data bytes
.............. ..
--- durer.home.com ping statistics
1015 packets transmitted, 1000 packets received, 1% packet loss
round-trip min/avg/max = 1.0/14.3/121.0 ms
```

Lo primero que es diferente de la salida estándar son las series de puntos (.....) mostradas. Esto indica que **ping** ha recibido respuestas de las solicitudes enviadas a `durer`. Segundo, nótese que de los 1,015 paquetes enviados, sólo se recibieron 1,000 paquetes. Y **ping** sólo reportó 1% de la pérdida de paquetes. Realmente, la pérdida de paquetes es más cercana a 1.5%. El comando **ping** reporta sólo números enteros, sin números a la derecha del punto decimal. En este caso, el problema no es significativo, ya que podemos determinar manualmente con más exactitud la pérdida, si es necesario. Sin embargo, esto es de alguna manera discutible, ya que la facilidad **ping** no es realmente considerada como la aproximación o el enfoque más científico y/o empírico para medir la pérdida de datos o el rendimiento de una red.

Quizás para muchos, la pregunta importante es: "¿Para qué usar la capacidad de inundación?" La razón es muy simple; intente producir una cantidad considerable de tráfico en la red y vea qué pasa. Encontrará que en el trabajo en red, determinar cómo funcionan algunas veces las cosas incluye intentar romperlas o hacer cosas de tal manera que excedan las limitaciones prácticas de un dispositivo o sistema. En el caso de los dispositivos de trabajo en red, como los enrutadores, los expertos en la industria y los usuarios quieren saber qué le pasaría a un dispositivo si se expone a patrones de alto tráfico. Medir el rendimiento y otras características de trabajo en red cuando la red tiene una carga excesiva ayudará al administrador de red a entender verdaderamente el comportamiento de dispositivos y sistemas de trabajo en red críticos, cuando están instalados en sus redes. La opción de inundación es una forma fácil de hacerlo.

La última opción interesante de **ping** es el modo silencioso. Esta opción `-q` se utiliza para limitar la cantidad de salida que se produciría normalmente. El comando

```
ping -c 100 didymus
```

produciría un gran número de líneas (más de 100) que representarían cada solicitud y respuesta asociada. Con la opción `-q`, sólo se presentan unas pocas líneas; el comienzo y el resumen:

```
ping  q -c 100 didymus
PING didymus-gw2.home.com (10.0.2.127): 56 data bytes

    didymus-gw2.home.com ping statistics ---
100 packets transmitted, 100 packets received, 0% packet loss
round-trip min/avg/max = 1.1/1.1/2.5 ms
```

Como se observa con claridad, esta opción reduce la salida, la cual puede ser justamente la que se necesita cuando se usa **ping** en una secuencia de comandos.

Herramienta tcpdump

Descripción general

El comando **tcpdump** es un seguidor de tráfico de red de propósito general que puede capturar y presentar paquetes y sus contenidos. Este comando puede usarse como un *analizador de protocolo*, proporcionando una de las mejores maneras de investigar los problemas de comunicación y/o conectividad entre sistemas y dispositivos de trabajo en red. La mayor parte del tiempo, la resolución de problemas de la red se enfoca en los problemas de configuración de la red y en diagnosticar las fallas relacionadas con el hardware. De vez en cuando, sin embargo, usted se enfrentará con un problema relacionado con el protocolo, y se verá forzado a escudriñar en el núcleo del protocolo en particular para resolver el problema. Con **tcpdump**, los paquetes explorados se mostrarán con información, ya sea en formato corto o largo, dependiendo de las opciones de línea de comandos que se usen. Además, **tcpdump** tiene un mecanismo de filtrado muy poderoso que puede buscar paquetes que correspondan a una cadena o un criterio específicos.

Tcpdump suministra dos modos principales de captura: promiscuo y no promiscuo. En el modo promiscuo, se captura cada paquete transmitido en la red, sea enviado o no el paquete al sistema en el cual **tcpdump** se está ejecutando. Éste es el modo, por ejemplo, que las sondas RMON usan cuando hacen seguimiento al tráfico de la red. Las sondas de red escuchan en la red el tráfico y recolectan la información y las estadísticas del protocolo. Debido a que los protocolos de red de área local (*Local Area Network*, LAN), como Ethernet, están basados en la difusión, cada marco que se transmite puede ser visto por cualquier interfaz de red conectada a la LAN. Cualquier dispositivo puede leer cada marco transmitido si ese dispositivo así lo decide y está configurado para hacerlo. Cuando un dispositivo o una interfaz lee cada marco de la red, se dice que está en modo promiscuo.

En la práctica, la interfaz debe configurarse para operación promiscua y sólo se usa en ocasiones especiales cuando se requiere el diagnóstico de red. Por lo anterior, sólo **root** puede habilitar el modo promiscuo en una interfaz. Ésta es la razón principal por la que a los usuarios diferentes de root no se les permite invocar **tcpdump**. Cuando un usuario no-root hace el intento de ejecutar este comando, se presenta el siguiente mensaje:

```
tcpdump: socket: Operation not permitted
```

Si se quiere dar a un usuario regular la habilidad de invocar el comando **tcpdump**, se puede usar el programa *setuid* para permitirle correr como **root** o instalar y configurar un programa como **sudo**. La utilidad **sudo** les da a usuarios específicos acceso a programas privilegiados, cuando el administrador del sistema lo considere apropiado. El apéndice C contiene información adicional acerca del comando **sudo**.

Con el modo no promiscuo, sólo estarán disponibles para la interfaz los marcos de difusión y los marcos dirigidos al sistema local. El término difusión (*broadcast*) realmente se refiere tanto a la difusión normal (con todos los 1s en el campo de destino) como al tráfico de multidifusión. En circunstancias normales, la interfaz está en modo no promiscuo.

Cuando se invoca **tcpdump** sin las opciones de línea de comandos, se abre la interfaz de red principal, y comienza a capturar los marcos desde la red local y a presentar sus contenidos. Debido a que **tcpdump** puede producir una cantidad significativa de salida, usaremos la opción silenciosa (`q`) para reducir la cantidad de salida presentada. Cuando se ejecuta por **root**, el comando

```
# tcpdump -q
```

mostrará todos los marcos de la red en formato de línea única, no explicativo (*non-verbose*). El formato de la salida incluirá un sello de tiempo, fuente y *hosts* destino (o direcciones), el protocolo de red de alto nivel, algunas banderas, información adicional del protocolo y un resumen:

```
tcpdump: listening on eth0
15:41:58.055268 durer.home.com.1032 > socrates.home.com.telnet: tcp 0 (DF)
[tos0x50]
15:41:58.055446 socrates.home.com.telnet > durer.home.com.1032: tcp 28
(DF)
```

```
15:41:58.274933 durer.home.com.1032 > socrates.home.com.telnet: tcp 0 (DF)
[tos0x50]
15:41:58.275115 socrates.home.com.telnet > durer.home.com.1032: tcp 164
(DF)
15:41:58.494694 durer.home.com.1032 > socrates.home.com.telnet: tcp 0 (DF)
[tos0x50]
15:41:58.494880 socrates.home.com.telnet > durer.home.com.1032: tcp 165
(DF)
15:41:58.544828 socrates.home.com > didymus-gw2.home.com: icmp: echo
request
15:41:58.545719 didymus-gw2.home.com > socrates.home.com: icmp: echo
reply5:34:10.601950 socrates.home.com.telnet > durer.home.com.1032: tcp
165 (DF)

8 packets received by filter
0 packets dropped by kernel
```

La salida incluye la hora, fuente/destino, puerto del protocolo, protocolo contenido en el marco e información adicional del protocolo. En este ejemplo, `durer`, el *host* fuente, y `socrates`, el *host* de destino, tienen establecida una sesión Telnet. Podemos decir esto sólo con mirar el puerto de destino, el cual es `telnet`. De manera predeterminada, **tcpdump** captura paquetes hasta que el usuario interrumpa el programa ingresando **^c.** Igualmente, `socrates` emitió una solicitud **ping** a `didymus-gw2`, y éste le devolvió una respuesta. La cadena `->` indica la dirección de la comunicación. Nótese que **tcpdump** siempre orienta la ruta de comunicación al punto a la derecha, como en el caso de la solicitud Echo ICMP anterior. Para indicar la comunicación en la otra dirección, **tcpdump** devuelve al *host* (no al puntero), como se muestra en la entrada de respuesta de Echo ICMP. El comando **tcpdump** presenta los protocolos de mayor nivel en minúsculas, como en `tcp` e `icmp`, seguidos de información más específica pertinente al protocolo, la cual podría incluir puertos, información adicional del protocolo y datos. La salida también incluye un resumen de la cantidad de paquetes obtenidos antes de que el usuario dé por terminado el comando.

El final de la salida incluye un conteo del número de paquetes capturados por **tcpdump** y el número de paquetes abandonados. En este caso, fueron capturados un total de ocho paquetes y fueron descartados cero paquetes.

La herramienta **tcpdump** proporciona un gran número de opciones de línea de comandos para seleccionar los modos de captura, controlar la salida, precisar las especificaciones del filtro y especificar las características adicionales de operación. Estas opciones se

agrupan de acuerdo con su función, e incluyen las siguientes categorías:

- modos de operación
- opciones de presentación
- opciones de filtro de paquete

Tabla 4.10 Opciones de modo de operación **tcpdump**.

Opción	Descripción
`-c`	Captura el número especificado de paquetes
`-F`	Usa el archivo como fuente para la expresión del filtro
`-i`	Captura los paquetes usando la interfaz de red alterna
`-p`	Inhabilita la captura en el modo promiscuo
`-r`	Lee el archivo de captura en lugar de la interfaz de red
`-w`	Guarda los paquetes sin procesar, para archivar

Opciones de modos de operación

Estas opciones se usan para controlar la manera como **tcpdump** capturará y presentará el tráfico de la red. Las opciones disponibles están resumidas en la tabla 4.10 y se describen completamente a continuación.

En general, **tcpdump** escuchará el tráfico en la interfaz de red primaria. Usualmente la interfaz primaria tiene el identificador numérico más pequeño, si el sistema contiene dos o más interfaces del mismo tipo. Por ejemplo, `eth0` se considera primaria cuando el sistema contiene dos interfaces Ethernet: `etn0` y `eth1`. Sin embargo, si se quiere correr **tcpdump** en una interfaz diferente, se utiliza la opción `-i` y el nombre del dispositivo, para especificar la interfaz alterna. Por ejemplo, para seleccionar la interfaz punto a punto (`ppp0`), se usa el siguiente comando:

```
# tcpdump -i ppp0
tcpdump: listening on ppp0
```

Como se indicó previamente, **tcpdump** capturará paquetes hasta que se ingrese **^c** desde el terminal de control (o si está en segundo plano, hasta que el proceso se termine con el comando **kill**). Si se desea especificar el número de paquetes a capturar, se emplea la opción `-c` seguida de un valor de conteo de paquete. Para capturar diez paquetes de la interfaz `eth1` se usa el siguiente comando:

```
tcpdump -t -q -i eth1 -c 10
tcpdump: listening on eth1
chips.home.com > didymus-gw2.home.com: icmp: echo request
didymus-gw2.home.com > chips.home.com: icmp: echo reply
chips.home.com > didymus-gw2.home.com: icmp: echo request
didymus-gw2.home.com > chips.home.com: icmp: echo reply
socrates.home.com.1032 > switch.home.com.snmp: udp 44
switch.home.com.snmp > socrates.home.com.1032: udp 111
socrates.home.com.1032 > switch.home.com.snmp: udp 51
switch.home.com.snmp > socrates.home.com.1032: udp 61
socrates.home.com.1032 > switch.com.snmp: udp 51
switch.com.snmp > socrates.home.com.1032: udp 54
socrates.home.com.1032 > switch.home.com.snmp: udp 51
```

En este caso, **tcpdump** capturó una sesión **ping** entre `socrates` y `didymus gw2`, tal como se detectó por las cuatro primeras líneas. Además, `socrates` consulta un dispositivo llamado `switch` usando SNMP, como se denotó por el puerto SNMP y el protocolo UDP que se usó (las líneas restantes). Se puede confirmar la cantidad de paquetes capturados al hacer el conteo del número de líneas presentadas. Especificar el número de paquetes a capturar es útil cuando se intenta hacer seguimiento a una transacción de red crítica que usa una cantidad fija de intercambios de paquetes. Esta opción es útil también cuando se hace seguimiento de paquetes en una secuencia de comando de shell, ya que no hay interés en detener **tcpdump** después de que éste ha arrancado. La opción `-t`, la cual remueve la información de sello de tiempo del paquete, se usó igualmente en este ejemplo.

Como se indicó antes, **tcpdump**, de manera predeterminada, abre la interfaz de red en modo promiscuo para capturar todo el tráfico de red. El modo promiscuo significa que todo el tráfico de red, independientemente del destino del paquete, será capturado por **tcpdump**. A veces, es más efectivo examinar los paquetes entregados a un *host* específico que leer todos los paquetes en la

red. Si queremos capturar aquellos paquetes dirigidos al *host* donde **tcpdump** está corriendo, la opción `-p` se usa para inhabilitar el modo de captura promiscuo. Más adelante veremos que se puede decir a **tcpdump** que capture los paquetes que vienen o van a un *host* en particular, usando filtros.

```
# tcpdump -d le1 -p
tcpdump: listening on eth0
```

Desafortunadamente, el comando **tcpdump** no confirma el uso de la opción `-p`. Como consecuencia, el usuario no tiene forma de saber, después de que el comando se ejecutó, con cuál modo está capturando, excepto si examina la salida para ver las direcciones destino (dirección de destino). En otras palabras, cualquier paquete que no es una difusión, o enviado a una dirección local, indica que **tcpdump** está capturando con el modo promiscuo habilitado. ¿Por qué se inhabilitaría el modo promiscuo con esta opción? A veces se vuelve necesario examinar sólo el tráfico que llega normalmente a un sistema en particular. Con esta opción, se puede ver cada paquete que esté destinado por el sistema local, y no más. Esto puede usarse para detectar con facilidad (haciendo seguimiento a los paquetes entrantes), por ejemplo, si una solicitud para un servicio particular está alcanzando el sistema. Dado un entorno cliente/servidor, se puede determinar si las solicitudes de los sistemas cliente están alcanzando al servidor (corriendo **tcpdump** en el servidor) o si un cliente en particular está transmitiendo solicitudes en primer lugar (corriendo **tcpdump** en el cliente). Es cierto que podemos usar filtros para lograr lo mismo, pero es más eficiente y fácil reducir el número de paquetes en primer lugar.

En una red muy activa o un sistema ocupado, y cuando se usen ciertas opciones de línea de comandos, **tcpdump** puede producir una gran cantidad de salida. Para hacer más fácil su manejo, se puede usar la opción `-w` a fin de redirigir la información capturada a un archivo. Una razón para usar un archivo es guardar la información capturada para una inspección y análisis posteriores. Esto podría incluir la manipulación de datos de otras maneras, posiblemente usando los datos para construir reportes especializados, y similares.

Para capturar el tráfico de red y guardarlo en un archivo llamado `traffic data`, se usa el siguiente comando:

```
# tcpdump -w traffic-data
tcpdump: listening on eth0
```

El archivo, `traffic-data`, se crea en el directorio local una vez se invoca el comando anterior. Sin embargo, los datos capturados no se escriben en el archivo hasta que el usuario interrumpa el programa o se use la opción `-c`. Una vez el usuario hace esto, se presenta lo siguiente:

```
48 packets received by filter
0 packets dropped by kerne
```

Nótese que el archivo creado no es un archivo de texto ordinario, sino que es un archivo de datos formateado especialmente que sólo entienden **tcpdump** y algunos programas más. Uno de tales programas, llamado **tcpslice**, puede cortar y unir diferentes archivos **tcpdump**.

Para aprender un poco más del archivo **tcpdump**, se puede usar el comando `file` UNIX. Si éste se corre junto con el archivo `traffic-data`, reporta que está asociado con el comando **tcpdump**, la versión del software, y que los datos contienen paquetes que son del tipo `Etnernet`, con una longitud de marco de 68 bytes. No es un archivo de texto legible para el ser humano. El comando UNIX

```
# file traffic-data
```

muestra lo siguiente:

```
traffic-data: tcpdump capture file (little-endian) - version 2.4
(Etnernet, capture length 68)
```

Como se mencionó previamente, es interesante que **tcpdump** etiquete los tipos de marco que están contenidos en los archivos de datos. En el ejemplo anterior, la etiqueta indica qué marcos `Etnernet` se almacenan en este archivo. Tipos adicionales incluyen `raw IP`, `FDDI` y otros. Así, si se hiciera una captura de un paquete en un enlace punto a punto, como en la interfaz `ppp0`, entonces se presentaría el tipo de paquete `raw IP`.

Si estuviéramos tratando de ver el archivo, aparecería una larga serie de caracteres extraños. Sin embargo, este archivo puede leerse en el programa `ethereal`, el cual proporciona un enfoque basado en GUI para la captura y el análisis de protocolo. El siguiente capítulo analiza esta herramienta.

El comando **tcpdump** hace un trabajo razonable de compresión de los datos almacenados en archivos. Por ejemplo, más de 3,000 paquetes ocupan aproximadamente 260K de espacio del disco. Si se va a guardar una cantidad significativa de datos capturados,

puede ser necesario comprimir los datos aún más usando **compress** o **gzip**. La utilidad **compress** se las arregla para reducir el tamaño de este archivo de datos a un poco más de 33K, mientras el programa `qzip` reduce el conteo en bytes aún más, a cerca de 18K. ¡Nada mal desde un tamaño en bytes de más de 260K!

Para presentar los paquetes guardados en un archivo de datos, se usa la opción `-v` seguida del nombre del archivo de captura de paquete. Por ejemplo, para repetir los paquetes guardados en el archivo `traffic-data`, se emite el siguiente comando:

```
# tcpdump  r traffic-data
```

Opciones de presentación

Estas opciones controlan cómo **tcpdump** presentará los paquetes de la red. La lista de las opciones disponibles en esta categoría se resume en la tabla 4.11 y se describe completamente a continuación:

Tabla 4.11 Opciones de presentación TCPDUMP.

Opción	Descripción
`-e`	Imprime la información de encabezado del nivel de enlace en cada línea
`-v`	Especifica el modo expresivo
`-q`	Especifica el modo rápido (*quick*); muestra poca información del paquete
`-t`	Inhabilita la impresión del sello de tiempo
`-s`	Limita el tamaño de los paquetes capturados
`-x`	Presenta tanto el formato hexadecimal como el ASCII

A veces es útil determinar la longitud de los marcos de enlace de datos, por ejemplo, cuando se investigan problemas de rendimiento relacionados con aplicaciones de más alto nivel. La herramienta **tcpdump** proporciona una forma de obtener el tamaño de cada marco, incluidas tanto la porción del encabezado como la de los datos, con la opción de línea de comandos `-e`. El siguiente comando y salida de ejemplo muestran una sesión de transferencia de archivos usando difusiones FTP y ARP:

```
# tcpdump -t -e
0:10:5a:28:5d:7c Broadcast arp 42: arp who-has didymus gw2.home.com
  tell socrates.home.com
0:10:5a:28:5d:7c Broadcast arp 42: arp who-has didymus-gw2.home.com
  tell socrates.home.com
0:10:5a:28:5d:7c Broadcast arp 42: arp who-has didymus-gw2.home.com
  tell socrates.home.com
0:10:5a:28:5d:7c 0:60:97:e:a3:6 1514: socrates.home.com.ftp data
durer.home.com.1036: tcp 1448 (DF) [tos 0x8]
0:10:5a:28:5d:7c 0:60:97:e:a3:6 1514: socrates.home.com.ftp-data
durer.home.com.1036: tcp 1448 (DF) [tos 0x8]
```

Como se puede observar, el campo longitud se presenta luego del protocolo, o si no se muestra el protocolo, después de la dirección destino. Muestra el tamaño total del marco en bytes. La razón por la cual los paquetes de `socrates` a `durer` son mayores que 1,500 bytes es porque FTP llena el paquete con tantos datos como pueda éste mantener. Ethernet tiene una capacidad de datos de aproximadamente 1,500 bytes, sin incluir la porción del encabezado del marco.

La opción `-x` proporciona una forma de presentar un vaciado hexadecimal de los marcos de red. Presenta la información del encabezado del nivel de enlace, como las direcciones de fuente y destino. Consideremos la serie de intercambios de paquetes cuando el *host* `rembrandt` trata de abrir una sesión FTP con un sistema llamado `durer`. El comando **tcpdump**:

```
tcpdump -d lel -x 0 tcp and port 21
```

capturará cualquier actividad FTP en la red. Cuando se ejecuta este comando y se arranca una sesión FTP, los paquetes se capturarán y presentarán de la siguiente manera:

```
0:60:97:e:a3:6 0:10:5a:28:5d:7c ip 78: durer.home.com.1044 >
socrates.home.com.f
tp: S 9262138:9262138(0) win 8192 <mss 1460,nop,wscale
0,nop,nop,timestamp[|tcp]
(DF) [tos 0x1d] (ttl 128, id 19970)
                        451d 0040 4e02 4000 8006 93c6 0a00 020a
                        0a00 02c9 0414 0015 008d 543a 0000 0000
                        b002 2000 a43e 0000 0204 05b4 0103 0300
                        0101 080a 0000
0:10:5a:28:5d:7c 0:60:97:e:a3:6 ip 74: socrates.home.com.ftp
durer.home.com.1044: S 108767057:108767057(0) ack 9262139 win 32120 <mss
1460,sackOK,timestamp> 1106589[|tcp]> (DF) (ttl 64, id 490)
                        4500 003c 01ea 4000 4006 2000 0a00 02c9
                        0a00 020a 0015 0414 40d4 a281 008d 543b
```

(*continúa*)

```
                        a012 7d78 92b7 0000 0204 05b4 0402 080a
                        0010 e29d 0000
0:60:97:e:a3:6 0:10:5a:28:5d:7c ip 66: durer.home.com.1044 >
socrates.home.com.ftp: .ack 1 win 8760 <nop,nop,timestamp 84320 1106589> (DF)
[tos 0x1d] (ttl 128,id 20226)
                    451d  0034 4f02 4000 8006 92d2 0a00 020a
                    0a00  02c9 0414 0015 008d 543b 40d4 a282
                    8010  2238 d35b 0000 0101 080a 0001 4960
                    0010  e29d
0:10:5a:28:5d:7c 0:60:97:e:a3:6 ip 163: socrates.home.com.ftp
durer.home.com.1044: P 1:98(97) ack 1 win 32120 <nop,nop,timestamp 1106595
84320> (DF) [tos 0x10] (ttl 64, id 493)
                    4510 0095 01ed 4000 4006 1f94 0a00 02c9
                    0a00 020a 0015 0414 40d4 a282 008d 543b
                    8018 7d78 b4a1 0000 0101 080a 0010 e2a3
                    0001 4960 3232
0:60:97:e:a3:6 0:10:5a:28:5d:7c ip 66: durer.home.com.1044
socrates.home.com.ftp: . ack 98 win 8663 <nop,nop,timestamp 84323 1106595> (DF)
[tos 0x1d] (ttl 128, id 20994)
                    451d 0034 5202 4000 8006 8fd2 0a00 020a
                    0a00 02c9 0414 0015 008d 543b 40d4 a2e3
                    8010 21d7 d352 0000 0101 080a 0001 4963
                    0010 e2a3
```

Nótese que alguna de la información no deseada removió manualmente de la salida, para hacerla más legible.

Como se puede observar, la salida muestra una línea de resumen que contiene la siguiente información (estamos usando el primer paquete del ejemplo anterior):

Dirección de la fuente de enlace de datos (*Datalink Source Address*)	`0:60:97:e:a3:6`
Dirección de destino de enlace de datos (*Datalink Destination Address*)	`0:10:5a:28:5d:7c`
Protocolo de alto nivel (*Highlevel Protocol*)	`ip`
Tamaño del marco (*Frame Size*)	`78`
Host fuente IP (*IP Source Host*)	`durer.home.com`
Puerto fuente TCP (*TCP Source Port*)	`1044`
Host Destino IP (*IP Destination Host*)	`socrates.home.com`
Puerto Destino TCP (*TCP Destination Port*)	`ftp`

Después del puerto destino TCP, se observan las banderas de paquete TCP. Estas transformaciones a tipos de paquete TCP estándar se listan en la tabla 3.5 del capítulo 3, "Conjunto de protocolos TCP/IP". La `S` indica que el paquete es un SYN, o arranca de una conexión TCP, mientras `P` significa que es un impulso de datos. **ack** indica un reconocimiento. La siguiente parte del listado muestra el marco completo en hexadecimal. Desafortunadamente, no hace un buen trabajo mostrándonos la versión ASCII del marco donde es apropiado. Por ejemplo, cuando un usuario arranca una sesión FTP hacia un *host*, se presenta con ella alguna información valiosa con un indicador de comandos de inicio de la sesión:

```
# ftp socrates
220 socrates.home.com FTP server(Version wu 2.4.2-VR17(1) Mon Apr
19 09:21:53 I
DT 1999) ready.
Name (socrates:root):
```

Esta información está contenida en los marcos que se mostraron anteriormente, pero **tcpdump** no muestra esta información en ASCII.

Uso de filtros de paquete

Un aspecto muy importante de la depuración del protocolo de red tiene que ver con el uso de filtros de paquete. Un filtro de paquete es un patrón predefinido que se compara con los paquetes entrantes, y consiste en una serie de una o más primitivas que pueden combinarse con operadores como `and`, `or` y `not`. Cuando el patrón encaja, ese paquete se captura y se presenta; o si no encaja, el paquete se descarta y no se presenta. Los filtros de paquete son útiles en la búsqueda de un tipo de protocolo particular o cualquier otra información específica disponible en el paquete. En el ejemplo anterior, primero usamos un filtro de paquete, diciéndole a **tcpdump** que estábamos interesados en presentar los paquetes que se relacionaban con una transferencia de archivo.

El comando **tcpdump** soporta los filtros de paquete definidos por el usuario. **tcpdump** instala un filtro cuando se da una expresión de filtro en la línea de comandos. Este filtro se coloca en memoria y filtrará de acuerdo con las reglas definidas, usando las palabras clave y las primitivas. Cada paquete se compara con el filtro, y cuando se encuentra uno igual, se presenta el paquete. En otro caso, el paquete se descarta. La figura 4.2 muestra una perspectiva de alto nivel de un filtro de paquete.

Normalmente, los paquetes de red se leen desde la interfaz de la red por el controlador asociado en nombre del kernel. Luego, **tcpdump** solicita la información desde el kernel usando llamadas del sistema. La herramienta **tcpdump** suministra un gran número de expresiones o primitivas predefinidas que pueden construir filtros de paquete muy potentes. Estas expresiones pueden dividirse en tres grupos. El primer grupo, llamado *primitivas de direcciones*, opera en paquetes usando la información de la dirección, como direcciones IP o de enlace de datos; el segundo grupo, conocido como *primitivas de protocolo*, opera en los paquetes que usan diferentes protocolos, incluidos IP, AppleTalk y otros. El último grupo incluye *primitivas aritméticas* (u operadores) y booleanas que pueden usarse con los primeros dos tipos de expresiones para producir filtros compuestos. Los filtros compuestos incluyen más de un tipo de expresión separados por palabras reservadas, como `and`, `not` y `or`. Estos filtros pueden

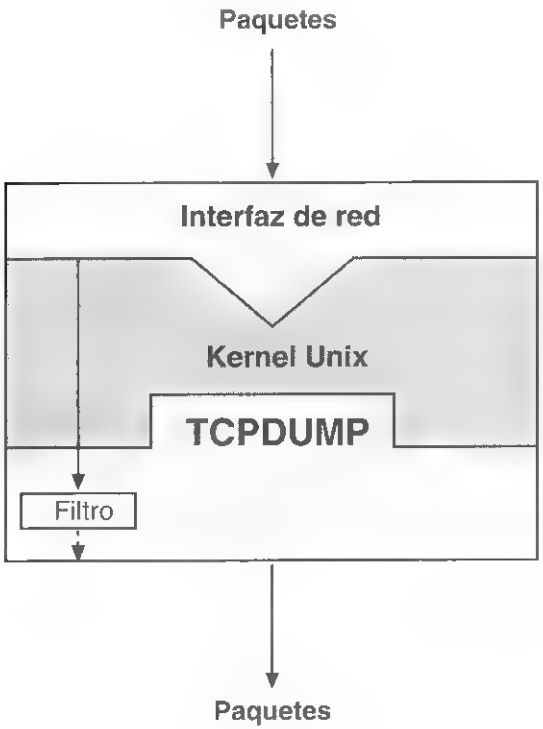


Figura 4.2 Diagrama del filtro de paquete **tcpdump**.

también incluir operadores aritméticos como +, -, * y otros. Estas expresiones son evaluadas como verdaderas o falsas, y el resultado determinará la acción que **tcpdump** llevará a cabo. A continuación sigue una descripción de las primitivas.

Primitivas de dirección

Las expresiones que operan en las porciones de direccionamiento de un paquete se listan en la tabla 4.12 y se describen con mayor detalle a continuación.

Tabla 4.12 Primitivas relacionadas con la dirección.

Primitiva	Descripción
`gateway`	Selecciona los paquetes usados por un *host* específico, actuando como puerta de enlace
`host`	Selecciona los paquetes de un *host* específico
`ipaddr` `etheraddr`	Selecciona paquetes de cualquier dirección IP o dirección de enlace de datos
`broadcast`	Selecciona los paquetes de difusión que contienen todos `1s` o `Fs` en cualesquiera campos fuente o destino
`multicast`	Selecciona paquetes que son enviados a direcciones de multidifusión
`net`	Selecciona paquetes que contienen porciones de red especificadas que se usan con el protocolo IP
`port`	Selecciona paquetes que contienen direcciones de puerto especificadas, que se usan con el protocolo IP

La primitiva `gateway` selecciona paquetes que han sido reenviados por un enrutador. Esto indica que la dirección de enlace de datos del paquete (ya sea fuente o destino) corresponde a la del enrutador, mientras que la dirección IP corresponde a la del *host*. Normalmente, un enrutador no cambiará la información de la dirección IP cuando reenvía paquetes, pero la dirección de enlace de datos corresponderá al dispositivo que reenvía el paquete.

La primitiva `host` seguida de un nombre de *host* válido puede seleccionar paquetes que sean enviados o recibidos desde el nombre de *host* especificado. La palabra clave **host** se usa principalmente para evitar la ambigüedad que pueda surgir si se fuera a especificar un nombre de *host* que resulta ser el mismo de una de las palabras clave existentes del **tcpdump**. Por ejemplo, cuando se hace seguimiento al *host* llamado `gateway`, debe usarse la palabra clave **host**, ya que la palabra **gateway** será interpretada como una palabra clave, en lugar de un nombre de *host* válido. Así, el comando **tcpdump** listado produce un error por que supone que la cadena gateway especifica un puerta de enlace local:

```
# tcpdump gateway
tcpdump: nostname required:
```

La forma de especificar la captura de paquetes desde un *host* llamado `cisco-gw2` sería la siguiente:

```
# tcpdump host cisco-gw2
```

Las opciones `ipaddr` y `etheradr` especifican las direcciones IP y direcciones de enlace de datos reales en formato punteado y de dos puntos, respectivamente. Por ejemplo, para capturar todos los paquetes de la dirección IP `10.0.2.100`, se usaría el siguiente comando:

```
# tcpdump ipaddr 10.0.2.100
```

Las primitivas `ipaddr` y `etneradr` corresponderán a la dirección de fuente o de destino. Algunas direcciones de enlace de datos comienzan con una letra y causaran que **tcpdump** las malinterprete como nombres de *hosts*, en lugar de direcciones verdaderas. Para evitar este problema, se inserta un cero enfrente cuando se especifican estos tipos de direcciones.

Para capturar paquetes de difusión, se usa la primitiva `broadcast` (difusión). Una `broadcast` es una dirección especial que indica que todos los dispositivos deben recibir el mensaje. Varios protocolos y servicios de red, como ARP, NIS y RIP, usan

difusiones para propagar información a lo largo de la red. Usar `broadcast` dará como resultado la captura de paquetes de difusión desde el nivel de enlace de datos, lo cual significa que será capturada cualquier dirección que contenga valores `255` o `FF` en el campo de fuente o de destino. Esto incluye los paquetes de enlace de datos que contienen difusiones (como las solicitudes ARP) y difusiones de protocolo de alto nivel (como la difusión IP). Esta primitiva se podría usar para capturar datos de enrutamiento del protocolo de información de enrutamiento (RIP), ya que los enrutadores difunden actualizaciones de enrutamiento periódicamente.

La primitiva `multicast` se emplea para obtener el tráfico de multidifusión, como radio de Internet. La dirección de multidifusión estándar de `224.0.0.1` soporta este tipo de tráfico, como se define por el estándar de multidifusión. Direcciones adicionales (tanto físicas como IP) se pueden usar en su sitio. Puede ser necesario determinar las direcciones de multidifusión exactas antes de empezar a filtrar estos tipos de paquetes.

Primitivas de protocolo

La aplicación **tcpdump** proporciona primitivas de protocolo como una forma abreviada de seleccionar tráfico de red específico, sin que sea necesario conocer la información de protocolo de bajo nivel. Por ejemplo, la primitiva `ip` puede usarse para capturar todo el tráfico IP. Sin esta palabra clave, se necesitaría usar el tipo IP de `x0800`, el cual es más difícil de recordar. Estas primitivas soportan TCP/IP, AppleTalk y la familia de protocolos DECnet. La tabla 4.13 lista y describe estas palabras claves de protocolo.

Para seleccionar una familia o un tipo de protocolos que no sea proporcionado directamente por **tcpdump**, se usa la primitiva `ethertype` junto con el código de tipo para el protocolo deseado. Por ejemplo, para hacer seguimiento a los paquetes Novell NetWare, cuyo código es del tipo `0x8137`, se usa el siguiente comando:

```
# tcpdump ethertype 0x8137
```

Nótese que como **tcpdump** no soporta directamente la familia de protocolo Novell, no se puede presentar información de paquete más allá de la capa de enlace de datos. Si **tcpdump** encuentra paquetes que contienen un encabezado Novell, sólo listará la información de enlace de datos. Sin embargo, a pesar de esta desventaja, **tcpdump** es útil todavía para identificar ciertos tipos de paquetes y suministrar información de conteo de paquete rudimentaria.

Tabla 4.13 Primitivas de protocolo.

Primitiva	Descripción
`apple`	Familia de protocolo AppleTalk
`arp`	Protocolo de resolución de dirección; incluye solicitud y respuesta
`fddi`	Protocolo de enlace de datos FDDI
`ethertype`	Otro tipo de protocolo (se usa con un código de tipo)
`decnet`	Familia de protocolo DECnet
`ip`	Protocolo de Internet
`icmp`	Protocolo de mensajes de control de Internet; incluye eco y respuesta
`rarp`	Protocolo de resolución de direcciones en sentido contrario (*Reverse Address Resolution Protocol*); incluye la solicitud y la respuesta
`tcp`	Protocolo de control de transmisión
`udp`	Protocolo de datagrama de usuario

Operadores

El comando **tcpdump** soporta varios tipos de expresión (u operador) y puede ser combinado con primitivas y calificadores para producir filtros compuestos. Estas expresiones incluyen los operadores aritméticos y booleanos listados en la tabla 4.14. Los operadores pueden construir expresiones potentes para buscar paquetes específicos. Las expresiones pueden estar compuestas de números, selecciones de campo de paquete, primitivas de longitud y operadores aritméticos. Para usar el valor de un campo en una expresión en un paquete, se usa la siguiente sintaxis:

```
primitive [offset [: size] ]
```

donde la palabra `primitive` se remplaza con `ether`, `ip`, `udp`, `tcp` o `icmp`. La palabra `offset` se usa en la base de la primitiva

de protocolo, y el *tamaño* (`size`) especifica la longitud del campo. Si no se suministra, se predetermina como número 1.

Las secciones de campo de paquete se pueden usar en una variedad de formas. Consideremos los siguientes ejemplos:

```
tcpdump "etner[1:4]&0xffffffff = 0xffffffff"
```

En este ejemplo, **tcpdump** presentará todos los paquetes de difusión transmitidos en la red local; es decir, todos los marcos con dirección destino de sólo unos (`255` en decimal, `0xff` en hexadecimal). El 1 en `ether[1:4]` indica el primer byte direccionable del marco (la dirección de destino), y el valor 4 especifica la longitud de este campo. A pesar de que las direcciones Ethernet son de seis bytes, se pueden examinar los primeros cuatros bytes para determinar si es una dirección de difusión.

Tabla 4.14 Operadores aritméticos y booleanos.

Operadores aritméticos	Descripción
>	Mayor que
<	Menor que
>=	Mayor o igual que
<=	Menor o igual que
=	Igual a
!=	Diferente
+	Más
-	Menos
*	Multiplica
/	Divide
&	Y a nivel de bits
|	O incluso a nivel de bits
^	O exclusivo a nivel de bits

Tabla 4.14 Operadores aritméticos y booleanos. (*continuación*)

Operadores booleanos	Descripción
`and` **or** &&	Concatenación
`or` **or** ||	Alternación
`not` **or** !	Negación

Para presentar todos los paquetes que se originan de un sistema Sun particular, por ejemplo, se usa este comando:

```
tcpdump "ether[6:4]&0xffffffff = 0x08002004"
```

Éste le dice a **tcpdump** que examine el sexto byte del marco (la dirección del marco fuente) y lo compare con las direcciones `0x08002004` usando el operador & (`y`). Esta dirección de enlace de datos representa la dirección de un sistema local llamado `monet`. Recuérdese que las direcciones Ethernet tienen seis bytes de longitud y que se pueden usar los primeros cuatro bytes para identificar al sistema deseado. Como resultado, se presentarán todos los paquetes transmitidos desde `monet`. Para identificar otro sistema, se obtiene la dirección de enlace de datos del sistema, se convierte en hexadecimal, y se coloca en el lado derecho del comando anterior.

Diversas primitivas

Unas cuantas primitivas adicionales que no pueden clasificarse como primitivas de dirección ni como primitivas de protocolo también están disponibles desde **tcpdump**. Éstas incluyen las siguientes:

- `greater` (mayor)
- `less` (menor)
- `length` (longitud)

Las primitivas `greater` y `less` se usan en conjunto con otros comandos **tcpdump** para filtrar, de acuerdo con la longitud total del paquete. Por ejemplo, para presentar todos los paquetes que son mayores que `56` bytes, se invoca el siguiente comando:

```
# tcpdump greater 56
```

Para presentar todos los paquetes que sean menores que 60 bytes, se usa la primitiva `less`:

```
# tcpdump -x less 60
```

Este comando presentará todos los marcos que sean menores que 60 bytes. Por ejemplo, los marcos de solicitud ARP (protocolo de resolución de direcciones) son menores que 60 bytes. Ejecutando este comando, presentará cualesquiera mensajes ARP, como se muestra aquí:

```
tcpdump: listening on eth0
23:45:29.240364 arp who-has durer.home.com tell socrates.home.com
0001 0800 0604 0001 0010 5a28 5d7c 0a00
02c9 0000 0000 0000 0a00 020a
```

¿Por qué se usarían los comandos **greater** o **less**? Principalmente para buscar paquetes con base en su tamaño, en lugar de su contenido. Otra primitiva, `length`, puede usarse también para manejar la captura de paquetes con base en su tamaño exacto. `length` puede usarse cuando la necesidad de capturar paquetes está basada en algún cálculo. Por ejemplo, el siguiente comando presentará cualquier paquete que no sea igual a 56:

```
# tcpdump length != 56
```

Calificadores

Tres calificadores pueden usarse en conjunción con las primitivas listadas en las tablas anteriores. Un calificador puede definir aún más las características de búsqueda exactas para un tráfico de red específico. Estos calificadores incluyen los siguientes:

- `from` (de) o `src` (fuente)
- `to` (para) o `dst` (destino)
- `etner`

Los calificadores `from` y `src` se usan con las primitivas `host`, `net`, `ipaddr`, `etheraddr`, `port` o `rpc` para filtrar con base en una dirección o puerto de destino específico. Los calificadores `to` o `dst` modifican las primitivas anteriores, pero darán como resultado los paquetes capturados que van a una dirección o un puerto particular. El modificador `etner` resuelve un nombre en relación con una dirección de enlace de datos cuando se usa con la primitiva *host*. Para ilustrar el uso de la palabra clave *host*, supongamos que queremos

capturar tráfico sólo de un *host* particular. Se podría usar el siguiente comando **tcpdump:**

```
# tcpdump from host monet
```

Contrastemos el comando anterior con el siguiente:

```
# tcpdump host monet
```

En el último ejemplo, se presentará todo el tráfico que involucra a `monet`, el cual incluye paquetes enviados a `monet` y recibidos por éste. En el otro ejemplo anterior, sólo se presentará el tráfico recibido de `monet`. Como se puede observar, esto puede hacer una gran diferencia cuando se intenta aislar un problema de red.

Ejemplos de tcpdump

La tabla 4.15 contiene una lista de ejemplos de comando y una descripción de cada uno. Se espera que estos comandos proporcionen una referencia rápida en el uso de **tcpdump** en situaciones del mundo real.

Herramienta traceroute

Descripción general

El comando **traceroute** examina y registra la ruta a un destino de red especificado. En una red IP tradicional, se usan uno o más enrutadores para proporcionar la conectividad entre diferentes redes IP. Los enrutadores IP vienen en diferentes formas y tamaños, desde un simple sistema UNIX multi homed con dos interfaces hasta una serie de enrutadores Cisco de capacidad industrial que contienen un gran número de interfaces. En cada uno de estos casos, la función de enrutamiento es principalmente la misma: reenviar paquetes IP desde una interfaz a otra con base en la información de enrutamiento establecida. El comando **traceroute** usa el campo tiempo de vida (TTL) contenido en un paquete IP e intenta obtener un mensaje ICMP `TIME EXCEEDED` desde cada *host* a lo largo de la ruta, hasta el destino. Unido a un intento de conectar al destino en un puerto que no se puede alcanzar, causará una respuesta sistemática de todos los enrutadores a lo largo del camino, hasta el último destino. Cumple esta tarea enviando solicitudes (o sondas) con un TTL de `1` y aumenta el TTL en 1 hasta que alcance el huésped deseado o exceda el valor TTL máximo. De manera predeterminada, el TTL es colocado en `30` saltos, pero esto se puede cambiar.

Tabla 4.15 Ejemplos del comando **tcpdump.**

Comando	Descripción
tcpdump host barney	Capturará y presentará el tráfico enviado desde el sistema llamado `barney` o entregado a éste. Este comando examina los campos de dirección fuente y dirección destino del encabezado IP
tcpdump host not barney	Capturará y presentará el tráfico de todos los *hosts* en la red, a excepción del sistema llamado `barney`
tcpdump host barney and host fred and\ (not pebbles\)	Capturará y presentará el tráfico de los *hosts* `barney` y `fred`, pero no del sistema llamado `pebbles`. Los paréntesis se adicionaron para que el comando fuera más legible. Sin embargo, cada uno de los operadores **tcpdump** aritméticos y booleanos tienen precedencia y los paréntesis pueden usarse para definir el orden de la interpretación del comando. Nota: como los paréntesis son especiales para ciertos shells, se deben cambiar por la diagonal inversa (*backslash*) (\)
tcpdump arp	Capturará y presentará todos los paquetes de protocolo de resolución de direcciones (ARP). Esto incluye las solicitudes y respuestas
tcpdump host durer and tcp	Capturará y presentará todos los paquetes de protocolo de control de transmisión (TCP) desde o hacia el *host* `durer`
tcpdump host vectra and port 23	Capturará y presentará todos los paquetes que estén usando el puerto `23` desde o hacia el *host* `vectra`. Esto significa inspeccionar todos los paquetes Telnet que van a este sistema desde otros en la red. Recuérdese, que el puerto `23` es el puerto del servicio Telnet para todos los paquetes que entran

Tabla 4.15 Ejemplos del comando **tcpdump**. (*continuación*)

Comando	Descripción
tcpdump ether multicast	Capturará y presentará paquetes de multidifusión. Véase el siguiente comando como alternativa
tcpdump 'ip[16] >=224'	Capturará y presentará todos los paquetes que usen la dirección de multidifusión. Este comando compara el decimosexto byte (el cual es una dirección de destino) del paquete IP con un valor de `224`. Éste prefija la dirección de multidifusión estándar de `224.0.0.1`, lo que incluye todos los *hosts* en el grupo de multidifusión predeterminado
tcpdump 'ether[0] & 1=1'	Capturará y presentará todos los paquetes de difusión. La secuencia `ether[0]` proporciona acceso al primer campo del campo de destino de enlace de datos Ethernet y se compara con el valor `1`. Si los campos destino contienen sólo `1`s (lo que sería cierto si la dirección de difusión se expresa en números binarios) y si el operador `and` (`&`) se aplica a un valor positivo, éste dará un valor de `1`. En este caso, la expresión es verdadera y los paquetes se presentan
tcpdump 'ip[2:2] >512'	Capturará y presentará todos los paquetes IP que sean mayores que 512 bytes. La secuencia `ip[2:2]` identifica el segundo byte del encabezado IP (el cual es el tamaño del paquete) y compara este valor de `512`. El `2:` indica el desplazamiento del paquete IP, mientras el `2` restante es el número de bytes en ese campo

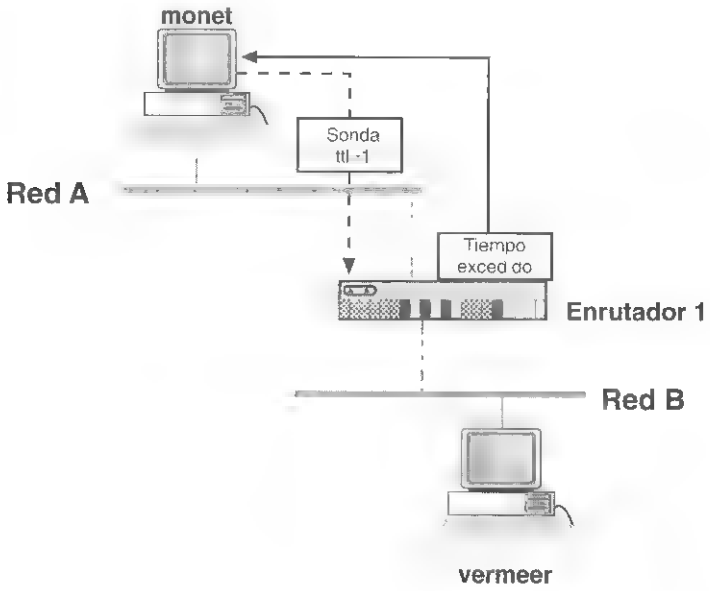


Figura 4.3 **traceroute** con un salto.

Este comando tiene un gran número de opciones de línea de comandos, pero el único argumento que se requiere es un nombre de *host* o una dirección IP de destino. Por ejemplo, para presentar la ruta IP entre el sistema local corriendo **traceroute** y el sistema destino llamado `vermeer`, se emite el siguiente comando:

```
# traceroute vermeer
```

Las figuras 4.3 y 4.4 presentan una red de muestra que consiste en un enrutador y dos nodos de red. Cuando el comando **traceroute** anterior se ejecuta en `monet`, se presentará la siguiente salida:

```
# traceroute vermeer
traceroute to vermeer (128.197.2.200), 30 nops max, 40 byte packets
1 router-1 (10.0.2.129) 4.256 ms * 2.899 ms
2 vermeer (128.197.2.200) 7.340 ms 7.433 ms 7.526 ms
```

De manera predeterminada, **traceroute** envía un total de tres sondas, cada una con un valor TTL diferente, para cada salto. La primera línea de la salida incluye el destino, junto con la dirección IP, el número predeterminado de saltos que usó **traceroute**, y el tamaño de los paquetes enviados. La segunda línea (con un 1) presenta el primer salto encontrado por **traceroute** (*véase* la figura 4.3).

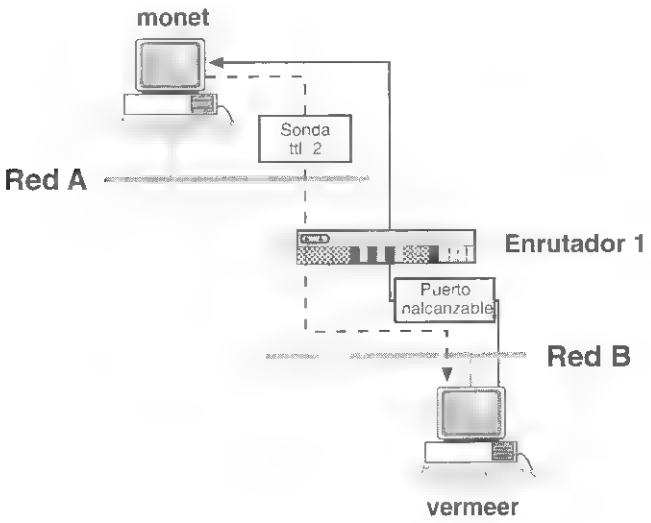


Figura 4.4 traceroute con dos saltos.

Debido a que `vermeer` está en una red física diferente de la de `monet`, debe usarse un enrutador para alcanzar este sistema. Como el enrutador predeterminado en este ejemplo es `router-1` (enrutador 1), el primer paquete se envía allá. El primer paquete enviado es un paquete de solicitud ICMP, con el campo TTL es tablecido en 1. Con IP, cualquier paquete que alcance al enrutador disminuye el TTL en 1, lo que equivale a cero. Cuando un enrutador obtiene un paquete y TTL es cero, se supone que descarta el paquete y le notifica al remitente. Esto obliga al enrutador a responderle a `monet` con un mensaje `TIME_EXCEEDED`. Después de que esto sucede, **traceroute** mide la cantidad de tiempo entre el momento en que envió el paquete y el momento en que obtuvo la respuesta. Esto se conoce como tiempo de ida y vuelta (*round-trip time*) o RTT y se presenta en milisegundos (una milésima de segundo), como se muestra después del nombre del *host* y la información de la dirección IP. Esto implica que el RTT de la primera serie de paquetes de sonda tomó `4.25` milisegundos (o 0.004 segundos), y la tercera serie tomó `2.89` milisegundos (o 0.028 segundos).

La segunda línea detalla el segundo salto de enrutamiento y muestra que **traceroute** alcanzó el sistema del destino `vermeer` con tiempos RTT más lentos que el primero (*véase* la figura 4.4). Cuando se envió la segunda sonda, el enrutador disminuyó el

TTL, y luego pasó este paquete a `vermeer`. Como **traceroute** está intentando tener acceso a un puerto no usado, `vermeer` responde con el error `PORT UNREACHABLE` (puerto no alcanzable). De hecho, como regla general en internets grandes (y a veces pequeñas), el rendimiento entre sistemas y redes podrá, y puede, variar considerablemente aun de un momento al siguiente.

No hay restricción en lo que constituye un destino; puede ser cualquier tipo de dispositivo en el rango que va desde un sistema de *host* simple hasta un enrutador de Internet. El único requisito es que debe soportar IP.

Lectura de la salida del traceroute

Debido a las variaciones y los problemas con los dispositivos enrutadores a lo largo de ciertas rutas, las fluctuaciones generales de las rutas de red pueden llevar a salidas de **traceroute** impredecibles y extrañas. Como consecuencia, podrían aparecer ciertos códigos después de que se presenta la información de temporización. La tabla 4.16 muestra estos códigos y sus significados.

Tabla 4.16 Códigos de presentación de **traceroute.**

Código	Significado
*	No hay respuesta a los paquetes de sonda
!	TTL en el paquete recibido se establece en 1
!H	El *host* destino no se puede alcanzar
!N	La red destino no se puede alcanzar
!P	El protocolo destino no se puede alcanzar
!S	La opción de la ruta fuente falló. En la práctica, esto no debe pasar, y si pasa, indica que un *bug* o un problema con el enrutador generó el error
!F	Se necesitó fragmentación del paquete de sonda. En la práctica, esto no debe suceder, y si sucede, indica que un *bug* o un problema con el enrutador generó el error

Tabla 4.16 Códigos de presentación de **traceroute**. (*continuación*)

Código	Significado
!X	La ruta está bloqueada debido a que la comunicación está siendo inhabilitada administrativamente. En otras palabras, la ruta está cerrada o bloqueada desde el punto de vista del software
!N>	Código de error ICMP donde N es ese número

Algunos de estos códigos de presentación están a la mano cuando usted depura problemas de la red. Por ejemplo, si un destino no se puede alcanzar (como *www.whitehouse.gov*) por el sistema que está corriendo el comando **traceroute**, ejecutar

```
monet# traceroute www.whitehouse.gov
```

producirá

```
traceroute to www.whitehouse.gov (198.137.240.91), 30 hops max, 40
   byte packets
1 monet (10.0.2.126) 4.281 ms .N * 1.896 ms !N
```

En este ejemplo, la red `198.137.240` no se puede alcanzar desde el sistema local, y **traceroute** presenta la cadena de error `!N` para indicar este problema. En este caso en particular, `monet` no puede enviar ningún paquete a `198.137.240`, ya que no existe una ruta para esa red. El * significa que el paquete particular de sonda nunca recibió una respuesta; el * se usa como un indicador de extemporaneidad. Para verificar más a fondo estos problemas, usemos el comando **ping** con el mismo destino. Así, el comando

```
monet# ping www.whitehouse.gov
```

mostrará el mismo problema, excepto que el error es un poco más descriptivo:

```
ICMP Net Unreachable from gateway monet (10.0.2.126)
for icmp from monet (10.0.2.126) to www.whitehouse.gov
```

De la misma forma, los códigos de error `.H` y `!P` se usan también para depurar problemas generales de red. Sin embargo, en estos dos casos, `!H` reporta cuándo un *host* no se puede alcanzar, y `!P` reporta cuándo el protocolo es inalcanzable. Se presentará el mensaje de *host* inalcanzable, por ejemplo, cuando la red tampoco se puede alcanzar.

Dado el hecho de que, en cualquier momento, el rendimiento o RTT entre redes y sistemas puede cambiar considerablemente, un rastreo al sitio Web del Museo de Louvre revela primero lo siguiente:

```
# traceroute 198.137.240.91
traceroute to 198.137.240.91 (198.137.240.91), 30 hops max, 40 byte
   packets
 1 10.0.2.76 (10.0.2.76) 19.906 ms 9.801 ms 8.174 ms
 2 199.70.195.38 (199.70.195.38) 197.460 ms 188.000 ms 181.029 ms
 3 12.77.194.1 (12.77.194.1) 166.802 ms 184.713 ms 185.857 ms
 4 12.127.12.205 (12.127.12.205) 245.026 ms 270.253 ms 266.718 ms
 5 12.127.15.145 (12.127.15.145) 215.191 ms 211.920 ms 208.979 ms
 6 192.205.31.165 (192.205.31.165) 217.875 ms 232.610 ms 222.274 ms
 7 204.6.117.65 (204.6.117.65) 266.797 ms 239.000 ms 215.671 ms
 8 38.1.4.69 (38.1.4.69) 235.431 ms 225.447 ms 301.119 ms
 9 38.1.25.5 (38.1.25.5) 235.369 ms 236.134 ms 263.557 ms
10 38.1.25.5 (38.1.25.5) 252.172 ms 238.984 ms 263.013 ms
11 38.146.148.45 (38.146.148.45) 241.956 ms 248.091 ms 243.300 ms
12 198.137.240.33 (198.137.240.33) 249.361 ms 228.717 ms 252.927 ms
13 198.137.240.91 (198.137.240.91) 238.799 ms 259.967 ms 236.384 ms
```

Cuando el rastreo se repite más tarde, muestra lo siguiente:

```
# traceroute 198.137.240.91
traceroute to 198.137.240.91 (198.137.240.91), 30 hops max, 40 byte
  packets
 1 10.0.2.76 (10.0.2.76) 7.619 ms 5.863 ms 6.206 ms
 2 199.70.195.42 (199.70.195.42) 177.685 ms 177.691 ms 177.842 ms
 3 12.77.242.129 (12.77.242.129) 170.712 ms 177.096 ms 173.517 ms
 4 12.127.12.205 (12.127.12.205) 260.239 ms 248.072 ms 252.829 ms
 5 12.127.15.145 (12.127.15.145) 219.767 ms 215.645 ms 232.399 ms
 6 192.205.31.165 (192.205.31.165) 232.259 ms 225.243 ms 219.236 ms
 7 204.6.117.65 (204.6.117.65) 228.997 ms 218.067 ms 219.365 ms
 8 38.1.4.69 (38.1.4.69) 445.758 ms 232.797 ms 276.249 ms
 9 38.1.25.5 (38.1.25.5) 245.674 ms 443.611 ms 577.309 ms
10 38.1.25.5 (38.1.25.5) 432.994 ms 222.527 ms 242.844 ms
11 38.146.148.45 (38.146.148.45) 257.668 ms 249.923 ms 263.074 ms
12 198.137.240.33 (198.137.240.33) 276.658 ms 242.361 ms *
13 198.137.240.91 (198.137.240.91) 248.266 ms 245.006 ms 251.071 ms
```

Como se puede observar, muchos de los tiempos de respuesta están muy cerca. Sin embargo, también hay demoras significativas con los saltos 8, 9 y 10. En estos casos, el RTT es casi el doble, lo que indica que el rendimiento en una red grande enrutada (como la de Internet) puede variar con el tiempo.

Cambio de las características operacionales

El comando **traceroute** contiene una cantidad de parámetros operacionales que pueden modificarse para afectar la manera cómo rastrea una ruta al destino especificado. Cada uno de estos parámetros tiene una opción de línea de comandos asociada que puede alterar sus valores predeterminados. La tabla 4.17 presenta estas opciones.

Tabla 4.17 Opciones operacionales **traceroute.**

Opción	Descripción
`-i`	Especifica una interfaz alterna
`-p`	Establece el puerto alterno para enviar paquetes de sonda
`-g`	Especifica un enrutador para enrutar fuentes extraviadas
`-f`	Establece el valor TTL inicial que debe usarse
`-s`	Usa la dirección especificada como la dirección fuente en la transmisión de paquetes de sonda
`-q`	Establece el número de consultas de sonda
`-m`	Establece los saltos máximos
`-d`	Habilita la bandera de depuración (`SO_DEBUG`)
`-F`	Especifica no fragmentar
`-t`	Establece la bandera de tipo de servicio (TOS)
`w`	Establece el tiempo de espera para los paquetes de sonda
`-x`	Especifica no calcular las sumas de verificación (*checksums*)

Cuando **traceroute** se ejecuta en un sistema que es multihomed (por ejemplo, un sistema que contiene más de una interfaz de red), selecciona la primera interfaz que encuentra. Desafortunadamente, esto podría ser lo que no se quiere porque al destino que usted está buscando sólo se puede tener acceso desde otra interfaz. Para abordar este asunto, puede usarse la opción `-i` a fin de forzar a **traceroute** a enviar paquetes de sonda usando la interfaz especificada con esta opción. Así, el comando

```
traceoute -i hem0
traceroute: Warning: Multiple interfaces found; using
10.0.2.126 @ hme0
```

hará que **traceroute** use la interfaz Fast Ethernet, en lugar de la interfaz FDDI que se usaría normalmente.

En ciertas situaciones, el valor TTL predeterminado de 30 no es suficiente para alcanzar un destino que podría contener un número grande de rutas. Cuando esto ocurre, **traceroute** no alcanzará nunca el destino. En esta situación, se usa la opción `-m` para aumentar el conteo de saltos.

Cuando se realiza un **traceroute** a ciertos dispositivos, podría fallar a veces, a pesar de que la ruta al dispositivo es operacional. Además, usando el comando **ping** junto con el dispositivo indicará que éste está trabajando correctamente. ¿Por qué? Antes de responder esta pregunta, miremos un ejemplo:

```
monet# traceroute -m 5 128.197.2.200
traceroute to 128.197.2.200 (128.197.2.200), 5 hops max, 40 byte packets
1 rodin (10.0.2.129) 10.193 ms * 2.158 ms
2 * * *
3 * * *
4 * * *
5 * * *
```

Este **traceroute** no produce respuesta a los paquetes de sonda. Nótese el uso de la opción `-m` para mantener pequeño el número de paquetes de sonda.

La respuesta a este problema se encuentra en el hecho de **traceroute** envía de manera predeterminada paquetes de sonda basados en el protocolo UDP. Es más probable que el destino en cuestión no soporte este protocolo directamente o no pueda manejar el puerto de destino UDP empleado. Desde un punto de vista TCP/IP, no todos los dispositivos requieren soportar UDP, y si lo soportan, no necesariamente soportan el número de puerto usado por **traceroute**.

Cuando **traceroute** envía paquetes de sonda, usa, de manera predeterminada, el puerto UDP de 33434 y supone que este puerto no está siendo usado por otra aplicación o servicio de red. Usa este alto número de puerto en espera de que el destinatario responda con un mensaje de puerto inalcanzable, terminando así el rastreo de la ruta. Por otro lado, si ese puerto está en uso, causará problemas para **traceroute**. Si esto sucede, se usa la opción `-p` seguida de otro número de puerto, y **traceroute** utilizará ese puerto en lugar del predeterminado.

```
monet# traceroute -p 10 -m 5 128.197.2.200
traceroute to 128.197.2.200 (128.197.2.200), 5 hops max, 40 byte
  packets
1 rodin (10.0.2.129) 10.193 ms * 2.158 ms
2 * * *
3 * * *
4 * * *
5 * * *
```

Si esto aún no da resultado, debe intentarse usar la opción `I`, la cual instruirá a **traceroute** para que use el protocolo ICMP en lugar del UDP cuando se envíen paquetes de sonda. Por tanto, el comando

```
monet# traceroute -m 5  I 128.197.2.200
```

con la opción `-I`, produce los resultados correctos:

```
traceroute to 128.197.2.200 (128.197.2.200), 5 hops max, 40 byte
  packets
1 rodin (10.0.2.129) 4.412 ms * 2.235 ms
2 vermeer (128.197.2.200) 6.875 ms 6.955 ms 6.935 ms
```

Como se puede observar, esto significó algo de ensayo y error para obtener el resultado deseado. Sin embargo, éste no es un ejemplo artificial; más bien, `vermeer` representa un dispositivo real: una impresora láser. El punto real aquí es que cuando se rastrea la ruta a un destino particular, puede haber muchas razones por las cuales **traceroute** falla en alcanzar un destino. La falla podría no significar que el dispositivo está abajo o desconectado de la red.

Opciones de presentación

Dos opciones están disponibles para modificar la salida de **traceroute**. La primera, `-v`, presenta, para cada salto, el tamaño y destino de los paquetes de respuesta. El siguiente es un ejemplo:

```
traceroute -v vermeer
# traceroute -i le1  v rembrandt
traceroute to rembrandt (10.0.2.75), 30 hops max, 40 byte packets
1 rembrandt (10.0.2.75) 56 bytes to 10.0.2.1 3.450 ms 2.085 ms
2.094 ms
```

La segunda opción, `-n`, presenta direcciones en forma numérica, en lugar de usar el nombre simbólico. Esto elimina la tarea agregada de **traceroute** de tener que resolver cada uno de los nombres de *host* de los enrutadores.

05

Utilidades y herramientas adicionales del sistema

Las herramientas que se presentan en este capítulo serán adiciones bienvenidas a la colección de utilidades y herramientas de software presentadas hasta ahora.

La mayor razón para incluir estas herramientas en un capítulo por separado es la siguiente: los comandos y programas analizados en el capítulo anterior están disponibles en diferentes sistemas UNIX, pero son generalmente parte del conjunto central estándar de ayudas de depuración. Por otro lado, estas herramientas no son tan ampliamente conocidas como las descritas en el capítulo 4. No obstante, estas herramientas suministran servicios muy importantes y tendrán un impacto significativo en el diagnóstico de temas y problemas del trabajo en red. Ellas incluyen:

Comando	Página
arpwatch	190
ethereal	194
fping	216
nmap	224
xtraceroute	246

Herramienta arpwatch

Descripción general

El comando **arpwatch** mantiene el rastro y los reportes de los cambios observados en las direcciones de enlace de datos en la red local. También rastrea los pares de direcciones de enlace de datos/IP, es decir, direcciones de Ethernet a protocolo de Internet (IP). Debido al crecimiento considerable de la cantidad de sistemas conectados a muchas redes, se ha vuelto primordial que un mecanismo esté disponible para registrar adiciones y cambios de dispositivos. Cuando **arpwatch** encuentra una discrepancia, registra la información en un archivo y envía un correo electrónico.

La tabla 5.1 presenta la pequeña cantidad de opciones de línea de comandos que soporta esta herramienta.

Una de las opciones más útiles es `-i`, la cual controla la interfaz que **arpwatch** usará cuando haga seguimiento a la red. De manera predeterminada, la interfaz primaria de red más baja en el inicio del sistema se selecciona para ser seguida por **arpwatch**. En un sistema con más de una interfaz, puede ser `etn0`, por ejemplo. Cuando **arpwatch** detecta cambios en las direcciones

Tabla 5.1 Opciones de línea de comandos **arpwatch**.

Opción	Descripción
`-d`	Habilita el modo de depuración
`-f`	Especifica un archivo alterno que puede usarse para almacenar la base de datos
`i`	Especifica otra interfaz de red que se sobrepone a la interfaz predeterminada definida en el sistema
`-r`	Analiza sintácticamente un archivo que contenga datos formateados por **tcpdump** para lectura, en lugar de hacerle seguimiento a la red

de enlace de datos, emite reportes específicos de mensaje. Los tipos de mensajes generados incluyen:

- nueva estación
- dirección Ethernet cambiada
- flip flop
- nueva actividad

Nueva estación

Este mensaje se utiliza cuando se ve por primera vez un nuevo sistema en la red. El software de **arpwatch** enviará un correo electrónico a la cuenta root en el sistema local, con un mensaje como el siguiente:

```
From root Mon Dec 27 13:05:09 1999
Date: Mon, 27 Dec 1999 13:05:09 -0400
From: arpwatch@socrates.home.com (Arpwatch)
To: root@socrates.home.com
Subject: new station

             nostname: probe
           ip address: 10.0.2.50
      ethernet address: 0:c0:d4:0:9:40
       ethernet vendor: Axon Networks Inc
             timestamp: Monday, Dec 27, 1999 13:05:08 -0400
```

El correo electrónico incluye el nombre del *host* (si está disponible), la dirección IP, la dirección Ethernet, el vendedor Ethernet, y sello de fecha/hora del evento. En este caso, se instaló una sonda RMON en la red local, y **arpwatch** detectó este nuevo dispositivo. También se envía un mensaje al proceso syslog que encontrará su camino dentro del archivo `/var/log/messages`. El formato del mensaje `syslog` es diferente del formato del correo electrónico que se muestra a continuación:

```
Dec 27 13:05:09 probe arpwatch: new station 10.0.2.50
0:c0:d4:0:9:40
```

Es claro que estar informado sobre este tipo de cambio en la red fue realmente útil. Los mensajes de nueva estación se generarán una vez que el dispositivo esté instalado y encendido. Después que este evento se ha registrado por primera vez, no se generarán mensajes adicionales, aun si el dispositivo es iniciado varias veces o si se desconecta de la red y se vuelve a conectar más tarde. Mientras la base de datos **arpwatch** se mantenga y preserve, un mensaje de nueva estación indica verdaderamente que un nuevo dispositivo se ha conectado a la red. Ésta es una excelente forma de ser informado cuando los usuarios agregan dispositivos a la red sin su conocimiento explícito.

Dirección Ethernet cambiada

Este mensaje se genera porque **arpwatch** detecta que la dirección de enlace de datos de un dispositivo cambió, pero la dirección IP sigue siendo la misma. Esto podría ocurrir cuando se instala una nueva tarjeta de interfaz para remplazar una unidad defectuosa o se cambian las direcciones de enlace de datos por cualquier otra razón, tal vez a propósito. El mensaje de correo electrónico que se genera incluye lo siguiente:

```
From root Fri Dec 24 23:07:43 1999
Date: Fri, 24 Dec 1999 23:07:43 -0400
From: arpwatch@socrates.home.com (Arpwatch)
To: root@socrates.home.com
Subject: changed ethernet address (switch.home.com)

            hostname: switch.home.com
          ip address: 10.0.2.60
    ethernet address: 0:0:1d:a:5d:1
     ethernet vendor: Cabletron
old ethernet address: 0:0:1d:a:5c:ff
```

```
old ethernet vendor: Cabletron
          timestamp: Friday, December 24, 1999 23:07:42 -0400
 previous timestamp: Thursday, December 23, 1999 21:25:30 -0400
              delta: 1 day
```

Realmente, este mensaje no fue el resultado de una nueva interfaz que se está instalando en un dispositivo. En su lugar, debido a que este dispositivo es un conmutador Cabletron Ethernet de múltiples puertos, contiene un gran número de interfaces de red. Como la conexión a la red para este dispositivo se movió de un puerto a otro, **arpwatch** detectó el cambio. La interfaz original tenía una dirección de `0:0:1d:a:5c:ff`, mientras el nuevo puerto tenía la dirección de `0:0:1d:a:5d:1`. Nótese que, en ambos casos, el vendedor fue identificado como `Cabletron`, que fue el vendedor correspondiente del hardware. Para hacer un poco más útil este reporte, se muestra el *delta* del tiempo transcurrido. Éste representa la cantidad de tiempo que ha corrido entre el instante en que **arpwatch** registró por primera vez el par enlace de datos/IP y el momento en que ocurrió por primera vez el cambio.

Flip Flop

Este tipo de mensaje indica que el host mencionado fue la dirección a la que más recientemente se le hizo seguimiento, pero ahora se ha convertido en la segunda dirección más vigilada. Por ejemplo, el mensaje

```
From root Sat Jul 31 14:07:15 1999
Date: Sat, 31 Jul 1999 14:07:15  0400
From: arpwatch@socrates.home.com (Arpwatch)
To: root@socrates.home.com
Subject: flip flop (durer.home.com)

            nostname: durer.home.com
          ip address: 10.0.2.10
    ethernet address: 0:60:97:e:a3:6
     ethernet vendor: 3Com
old ethernet address: 0:0:1d:a:5c:ff
 old ethernet vendor: Cabletron
           timestamp: Saturday, July 31, 1999 14:07:15 -0400
  previous timestamp: Saturday, July 31, 1999 14:06:29  0400
               delta: 46 seconds
```

indica que el *host* más recientemente visto se llama `durer.home.com`, el cual contiene la dirección de enlace de datos de `0:60:97:e:a3:6` y utiliza una tarjeta de interfaz 3Com. La segunda dirección más vista fue el dispositivo Cabletron, con la dirección de enlace de

datos `0:0:1d:a:5c:ff`. Antes de que este mensaje se emitiera, el dispositivo Cabletron era el más activo en la red.

Nueva actividad

El mensaje `new Activity` (*nueva actividad*) es generado cuando un par de direcciones Ethernet/IP se ve por primera vez en un periodo de seis meses (o más). Esto será útil para rastrear dispositivos que por alguna razón han estado fuera de servicio o desconectados de la red por mucho tiempo, y ahora se ponen de nuevo en servicio o se vuelven a conectar a la red. En la práctica, sin embargo, éste puede no ser un mensaje importante, debido a la duración. En su lugar, sería mejor si el intervalo de tiempo pudiera ser ajustado por el usuario para que fuera menor. No obstante, en la versión actual de la herramienta, esta característica no está soportada.

Una de las características más interesantes de la herramienta incluye el análisis sintáctico de la salida del **tcpdump**. Al hacer uso de la opción `-r`, **arpwatch** analiza sintácticamente estos archivos formateados de manera especial, construye una base de datos y reporta la historia de la red con los mensajes de seguimiento analizados previamente. Esto puede ser útil cuando se necesiten reportes **arpwatch**, pero por alguna razón **arpwatch** no puede ser ejecutado en tiempo real para hacerle seguimiento a la red.

Herramienta ethereal

Descripción general

El comando **ethereal** es una solución muy potente para hacer seguimiento de red Ethernet que puede usarse para capturar y analizar el tráfico de red en modos de tiempo real o fuera de línea. Esta herramienta capta dónde termina **tcpdump**, proporcionando una completa decodificación de protocolos a nivel de paquete que cuenta con el soporte para muchos protocolos de red populares y una interfaz de captura de paquete fácil de usar. Además, **ethereal** soporta muchos otros formatos de captura de paquetes comerciales, que brinda la oportunidad de continuar usando el software existente y procesar estos archivos de datos con **ethereal**.

Esta herramienta proporciona una interfaz GUI robusta que hace más eficiente la captura de los datos de red y el subsiguiente examen de tráfico de red. La ventana principal contiene un conjunto de elementos de menú, tres paneles, más tres estados o cuadros de entrada en la parte inferior, como se muestra en la figura

5.1. Muchos de los elementos del menú se explican por sí mismos y no se cubrirán en detalle. El primer panel contiene un resumen de todos los paquetes capturados. Este panel contiene campos de número de paquete, sello de tiempo, fuente, destino, protocolo e información. Como se verá más adelante en esta sección, estos campos se pueden personalizar, y el usuario puede remover o incluir campos de información adicional del paquete.

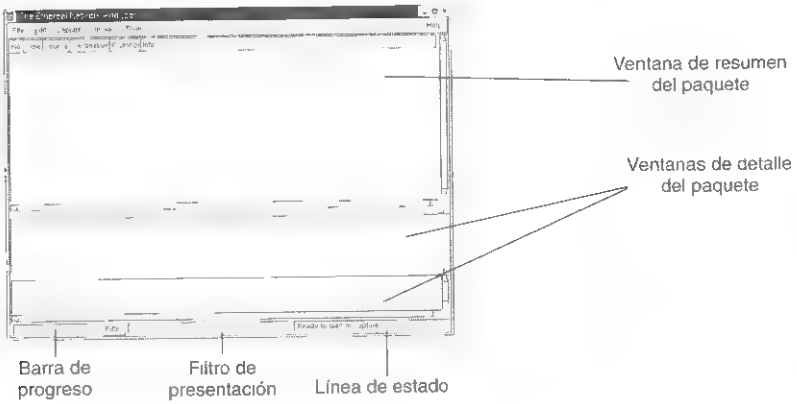


Figura 5.1 Ventana principal **ethereal.**

El siguiente panel muestra una vista jerárquica (detalle del paquete) que contiene todos los protocolos de capa superior presentados en un formato de árbol. El panel restante incluye una presentación hexadecimal de todo el paquete. A medida que el usuario se mueve a través del segundo panel, por medio de la vista de árbol, el panel inferior se resalta donde el cursor está posicionado en el paquete. Una barra de progreso en el lado izquierdo de la ventana principal muestra el estado de las tareas, y se suministra un cuadro de texto filtro para aceptar el filtrado de los paquetes presentados en el primer panel. Finalmente, un cuadro de información en el lado derecho de la ventana principal indica el nombre del archivo de datos y el número de paquetes abandonados.

El software soporta varias opciones importantes de línea de comandos, que pueden ser divididas en dos categorías: opciones de modo de presentación y opciones de modo de función. Las opciones de modo de presentación pueden controlar elementos como el tamaño de los paneles individuales de presentación y el tipo y tamaño de la fuente que se utiliza. La tabla 5.2 incluye las opciones de presentación.

Tabla 5.2 Opciones de presentación **ethereal**.

Opción	Descripción
`-B`	Controla la altura inicial del panel hexadecimal de la parte inferior
`-P`	Controla la altura inicial del panel de resumen de paquetes
`-T`	Controla la altura inicial del panel de vista de árbol de los paquetes
`-b`	Usa la fuente especificada en negrilla para la presentación del campo del paquete
`-m`	Usa la fuente especificada, en lugar de la predeterminada
`-t`	Establece el formato del sello de tiempo del paquete cuando se presenta en el panel del paquete de resumen

Las opciones de modo funcional controlan elementos como cuál interfaz de red se utilizará para leer el tráfico capturado, el tamaño máximo de los paquetes que debería ser leído durante la captura de paquetes, etc. La tabla 5.3 presenta estas opciones de comando.

Tabla 5.3 Opciones funcionales **ethereal**.

Opción	Descripción
`c`	Captura el número de paquetes tal como se haya especificado con esta opción
`-h`	Muestra la versión e información
`-i`	Especifica la interfaz de red para capturar paquetes
`-n`	Inhabilita la resolución del nombre del *host* y las transformaciones del nombre del puerto TCP/UDP

Tabla 5.3 Opciones funcionales **ethereal**. (*continuación*)

Opción	Descripción
`-r`	Obtiene los datos del tráfico de red del archivo especificado
`-s`	Especifica la longitud predeterminada del paquete cuando se captura tráfico de red activa
`-w`	Especifica el archivo de captura que se va a usar cuando se guardan datos en el disco

Muchas de las opciones funcionales son las mismas del comando **tcpdump**, las cuales fueron analizadas en el capítulo anterior. Como tales, muchas de las opciones deben ser familiares en este momento para el lector. No obstante, la opción `-s` merece un poco más de explicación. Por defecto, **ethereal** intenta leer todo el paquete mientras captura tráfico en la red. Sin embargo, con esta opción es posible limitar el número real de bytes que **ethereal** leerá de cada paquete. La longitud predeterminada de bytes que **ethereal** lee es `65535`, la cual es suficientemente grande para manejar muchos, si no todos, los paquetes que **ethereal** puede encontrar. Controlar el tamaño del paquete tiene un impacto en el rendimiento, el almacenamiento de datos y el análisis del paquete.

Los protocolos de enlace de datos de capa inferior de trabajo en red, como Ethernet y FDDI, tienen marcos de tamaño fijo que representan la cantidad máxima de datos que se pueden llevar en cada marco. Los marcos pueden ser más pequeños que el tamaño máximo. La tabla 5.4 muestra estos tamaños. Como se puede observar, los tamaños varían entre los protocolos, pero lo importante para recordar es que, sin importar el tamaño de los protocolos de nivel superior, como IP, lo que define el tamaño del marco físico es el protocolo de enlace de datos, no los de alto nivel. Además, muchos protocolos de capas superiores, como TCP y otros, tienen la capacidad de especificar la longitud de sus paquetes. En muchos casos, usualmente hay un techo o tamaño máximo permitido. Para IP, el máximo es `65535` y resulta ser el mismo del límite de **ethereal**.

Con esta información se puede ajustar arbitrariamente la cantidad de datos leídos durante la captura, en lugar de leer todo el marco. ¿Por qué hacer esto? Bueno, para principiantes, si se está interesado sólo en unos cuantos campos de paquete, como fuente, destino y tipo de marco, existe una pequeña razón para procesar el marco, con el fin de incluir campos adicionales. Desde un punto de vista de rendimiento, es mucho más fácil procesar una pequeña cantidad de bytes por marco. Igualmente, cuando se filtran y/o guardan datos en un archivo UNIX, será mucho más rápido. Por último, en un esfuerzo por mejorar el proceso de captura o reducir la cantidad de datos que se van a manejar, sacrificamos funcionalidad. Cuando se pone un límite artificial en el tamaño del marco, se afecta la cantidad de protocolos de alto nivel disponibles para esta revisión y análisis. Por ejemplo, cuando se ejecuta el comando

```
# ethereal -s 64
```

arranca **ethereal** y cambia el tamaño predeterminado del paquete a 64 bytes. Éste es el tamaño mínimo del marco de Ethernet, como se muestra en la tabla 5.4. Esta tabla incluye también tamaños adicionales del protocolo de enlace de datos. Si se está interesado en examinar los datos TCP, posiblemente de una sesión FTP, usar este tamaño no permitirá hacerlo; truncará los marcos y la información FTP se perderá. El mejor enfoque es usar el tamaño predeterminado, a menos que se estén capturando muchos paquetes y se deba reducir la cantidad total de datos guardados. La regla general es que si se va a capturar menos que lo predeterminado, se limitará la cantidad de protocolos de alto nivel que pueden ser decodificados en última instancia desde los paquetes capturados.

Tabla 5.4 Tamaños de marcos de enlace de datos.

Protocolo de enlace de datos	Mínimo	Máximo
ATM	6	53
Ethernet	64	1518
FDDI	64	4500
Fiber Channel	0	2112
TokenRing	48	2048

Captura del tráfico de red

Para empezar a capturar información de la red, se da clic en el menú de `Capture` (*Captura*), y luego se selecciona el elemento del submenú `Start... Ctl+k` (*Inicio... Ctl+k*). Como otra alternativa, se puede emitir un **^k** (**Control k**) del teclado. Una vez hecho esto, se presentará la ventana de Preferencias de Captura **ethereal**, como se muestra en la figura 5.2.

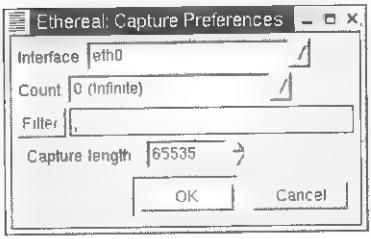


Figura 5.2 Ventana de Preferencias de Captura **ethereal**.

Esta ventana le permite al usuario cambiar la interfaz de red, el conteo de paquete y la longitud de marco, antes de que comience la sesión de captura. Además, el usuario puede definir un filtro en el cuadro de texto. Recuérdese que en el capítulo anterior se mencionó que un filtro puede usarse para clasificar completamente el tráfico de red y recoger tipos específicos de marcos o cualquier información que pueda concordar con un patrón numérico y/o de carácter. La herramienta **ethereal** soporta las primitivas y las palabras clave del filtro de protocolo **tcpdump**. Como consecuencia, filtros de mucha potencia pueden definirse muy rápida y fácilmente.

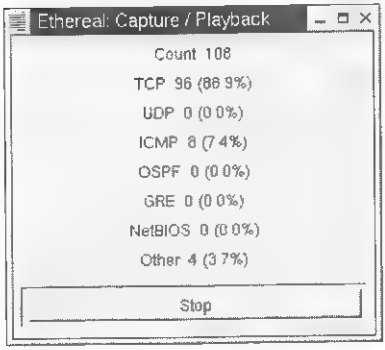


Figura 5.3 Ventana de Captura/Playback **ethereal**.

Al dar clic en el botón OK comenzará la captura. Al hacerlo, desaparecerá la ventana de Preferencias de Captura **ethereal** y aparecerá la ventana de Captura/Playback **ethereal**, como se muestra en la figura 5.3.

Esta ventana muestra que está corriendo un conteo de marcos obtenido por **ethereal**, clasificado por tipo de protocolo y porcentaje del número total de marcos recibidos. La figura 5.2 muestra un total de 108 paquetes capturados, de los cuales 96 son TCP, ocho, ICMP, y los cuatro restantes, considerados de *otro* (*other*) tipo. *Other* significa paquetes capturados de la red que no concuerdan con los tipos de paquete listados en la ventana de conteo. Esto proporciona una retroalimentación visual de las actividades de **ethereal** y una toma instantánea rápida de cuáles tipos de marcos han visto en la red. Si se desea detener la captura, se da clic en el botón Stop. A veces, puede verse cierto retraso desde el momento en que el usuario da clic en el botón STOP hasta el momento en que la aplicación realmente detiene el tráfico de captura. Esto, en parte, dependerá de la carga de la red u otros procesos que corren en el sistema. Además, si se definen filtros, esto puede hacer que **ethereal** sea un poco más lento cuando responda al control de la tecla y/o el ratón o *mouse*.

Una vez **ethereal** ha detenido la captura de datos, presenta todos los marcos en el primer panel de resumen de paquetes. La figura 5.4 muestra un ejemplo representativo de cómo lucirá la ventana principal después de capturar el tráfico de red.

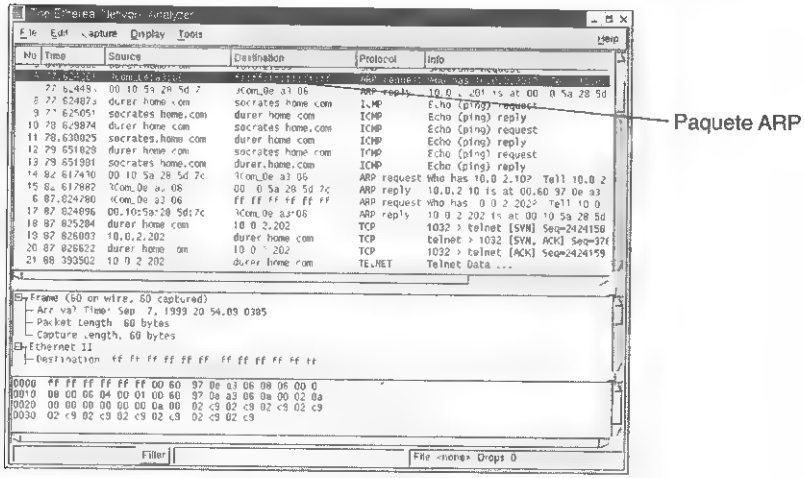


Figura 5.4 Paquete ARP seleccionado, en detalle.

En este ejemplo, se tienen diferentes tipos de paquete, como lo indica el campo `Protocol`. Se pueden ver paquetes `ARP`, `ICMP`, `TCP` y `TELNET`. Si se quiere examinar un único paquete en mayor detalle, simplemente se da clic en él. En la figura 5.4, el primer paquete `ARP` fue resaltado con el ratón, haciendo que los dos paneles inferiores presenten información adicional del paquete. Como se estableció previamente, el panel del medio contiene una vista de árbol de cada una de las secciones principales del marco. Todos los contenidos del panel se muestran a continuación, sin las marcas jerárquicas que se muestran en el panel de la mitad:

```
Frame (60 on wire, 60 captured)
    Arrival Time: Dec 27, 1999 20:54:09.0385
    Packet Length: 60 bytes
    Capture Length: 60 bytes
Ethernet II
    Destination: ff:ff:ff:ff:ff:ff (ff:ff:ff:ff:ff:ff)
    Source: 00:60:97:0e:a3:06 (3Com_0e:a3:06)
    Type: ARP (0x0806)
ARP request
    Hardware type: Ethernet
    Protocol type: IP
    Hardware size: 6
    Protocol size: 4
    Opcode: 0x0001 (ARP request)
    Sender hardware address: 00:60:97:0e:a3:06
    Sender protocol address: 10.0.2.10
    Target hardware address: 00:00:00:00:00:00
    Target protocol address: 10.0.2.201
```

¡El listado anterior incluye sólo un paquete! Éste representa la solicitud Echo ICMP enviada desde la interfaz de red `3Com_0e:a3:06` (la cual es la tarjeta de interfaz del sistema `durer`). Nótese cómo remplaza **ethereal** los tres primeros bytes de la dirección de hardware con una cadena que indica claramente el fabricante de la interfaz.

Cada paquete contiene por lo menos tres secciones. Éstas incluyen `Frame`, `Ethernet II` y `ARP`. La sección `Frame` representa el número de marco, el cual indica el conteo o posición en la lista de paquetes capturados. Este paquete es el sexagésimo marco leído desde la red. También se muestran el campo `Arrival Time`, el campo `Packet Length` y el campo `Capture Length`. A menos que los tamaños de los marcos fueran limitados específicamente en la línea de comandos o se use el diálogo de captura, la longitud de paquete y las longitudes capturadas deben ser iguales.

El campo `Ethernet II` representa el marco de enlace de datos, el cual es formateado de acuerdo con la especificación Ethernet. Recuérdese que cada marco Ethernet contiene un encabezado de longitud fija que incluye preámbulo, destino, fuente, tipo, datos e información de suma de verificación. Sin embargo, **ethereal** sólo reporta los campos `Destination` (*Destino*), `Source` (*Fuente*) y `Type` (*Tipo*). Los otros campos no se presentan, ya que el controlador Ethernet de bajo nivel no facilita la información.

La sección restante incluye la porción `ARP`, la cual indica, en general, los protocolos de más alto nivel que han sido encapsulados en la porción de datos del marco Ethernet. Como se indicó previamente, muchos protocolos de más alto nivel –como TCP, IP e IPX– pueden ser encapsulados en un marco igual a ARP. En este caso, el paquete ARP incluye información específica relacionada con el protocolo ARP: tipo de hardware, información de la dirección de hardware, tipo de paquete ARP, información de la dirección de hardware del remitente, y la dirección de hardware objetivo (la deseada). Podemos determinar que este paquete es una solicitud ARP, ya que el campo `opcode` está establecido en 1 (o `0x0001`) y la dirección de hardware objetivo (`Target hardware address`) es nula (o `00:00:00:00:00:00`).

El panel de la parte inferior contiene el formato hexadecimal del paquete, presentado en filas y columnas. A medida que el usuario tenga acceso a cada uno de los campos en el panel de la mitad, los bytes que representan estos datos se resaltan en este panel inferior. Por ejemplo, cuando un usuario selecciona el campo `Source` Ethernet del paquete, se presentará lo siguiente:

```
0000 ff ff ff ff ff 00 60 97 0e a3 06 08 06 00 01
.....` ....  ..
0010 08 00 06 04 00 01 00 60 97 0e a3 06 0a 00 02 0a
.. ....  ........
0020 00 00 00 00 00 00 0a 00 02 c9 02 c9 02 c9 02 c9
........ ........
0030 02 c9 02 c9 02 c9 02 c9 02 c9 02 c9
........ ...
```

Como se indicó, el conjunto de valores `00 60 97 0e a3 06` se muestra en negrilla para representar la dirección de enlace de datos del remitente de este paquete. Los datos en este panel son muy útiles, ya que proporcionan una conversión rápida al formato de bajo nivel, la cual puede usarse para identificar los patrones de tráfico de red. El formato hexadecimal es casi universal

cuando se representan paquetes en este nivel. En el último capítulo, se muestra cómo usar **tcpdump** para explorar una cadena específica en un paquete. Los datos representados en el formato hexadecimal proporcionan una referencia rápida que se necesite para filtrar cualesquiera patrones específicos.

Como **ethereal** soporta archivos formateados específicamente por **tcpdump**, cualquier archivo generado en este formato (como el producido con la opción `-w` de **tcpdump**) puede ser leído y procesado por **ethereal**. De hecho, ambos programas son completamente compatibles; la salida generada por **ethereal** puede ser procesada igualmente por **tcpdump**.

Uso de filtros

El software de **ethereal** suministra dos tipos de filtros: el filtro de captura de paquete y el filtro de presentación. Como se indicó, **ethereal** soporta las palabras clave de filtro, operaciones y sintaxis, de manera similar al comando **tcpdump** que se analizó en el capítulo anterior. Los filtros de captura pueden establecerse antes de que haya comenzado una sesión de captura, especificando uno de ellos en el cuadro de texto filtro, como se muestra en la figura 5.5. Por ejemplo, la figura muestra un filtro que capturará sólo paquetes TCP usando la cadena de filtro `tcp`. Este filtro es conocido como un filtro *ad-hoc* o temporal, ya que, una vez se ha terminado la aplicación, se pierde la definición de filtro. La cadena TCP es una palabra clave usada por **ethereal**, la cual filtrará todos los paquetes que usen TCP.

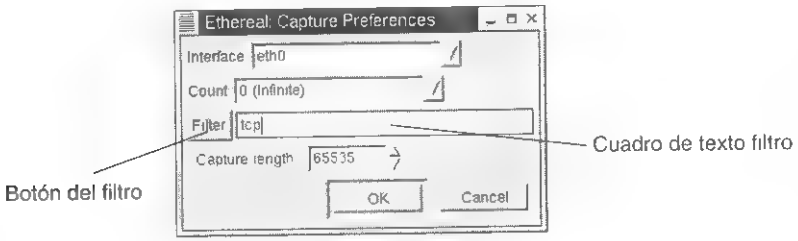


Figura 5.5 Ventana de filtro de paquete TCP.

También pueden crearse filtros permanentes. Éstos estarán disponibles a través de las sesiones de captura de paquete. Para crear este tipo de filtro, se da clic en el botón de filtro. Cuando se haya hecho esto, se presentará el cuadro de diálogo `Etnereal:Preferences`, como se muestra en la figura 5.6. Ésta

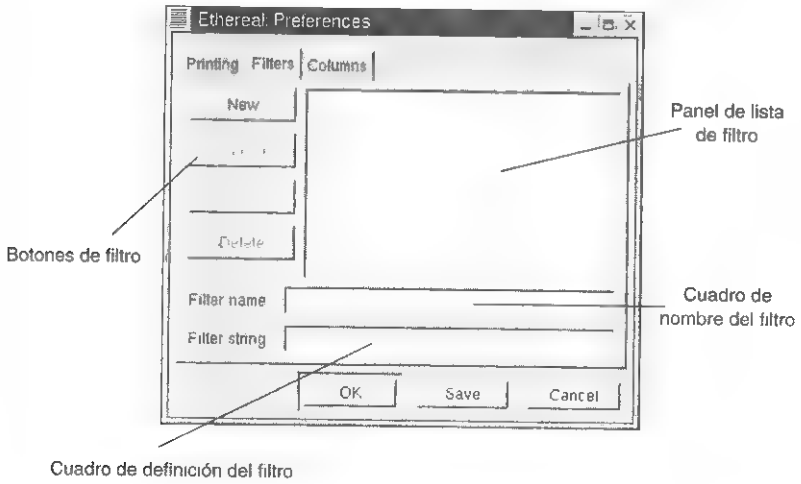


Figura 5.6 Ventana de filtro **ethereal.**

es la misma ventana que se presenta cuando el usuario tiene acceso a las opciones **ethereal.**

Por defecto, **ethereal** no proporciona definiciones de filtro existentes de manera predeterminada. En su lugar, el usuario debe crearlas usando este cuadro de diálogo y guardarlas para uso futuro.

Ésta es la razón por la cual el panel de definición de filtro está vacío. Si el usuario hubiera definido filtros, éstos se presentarían.

La ventana de filtro contiene un conjunto de botones, cuadros de entrada de texto y un panel de definición de filtro. Para construir un nuevo filtro, se escribe el nombre del filtro en el cuadro de texto `Filter name:` (Nombre de filtro:). En este ejemplo, se usa el nombre `ip broadcast`, el cual se describió en el capítulo anterior. Recuérdese que este filtro inspeccionará el encabezado IP del protocolo IP, para determinar si el paquete usa una dirección de difusión IP. Luego, se ingresa la definición del filtro en el cuadro de texto `Filter string` (Cadena de filtro); se usa la cadena: `ip[19]&0xff - 0xff`. Luego se da clic en el botón New. En este momento, el filtro `ip broadcast` se agregará al panel de lista de filtros. Finalmente, se da clic en el botón `Save` para guardar permanentemente el filtro, para uso futuro. Después de que se han completado estos pasos, la ventana de filtro debe contener la definición de filtro que se acabó de crear, como se muestra en la figura 5.7.

La ventana también contiene dos definiciones adicionales de filtro que fueron creadas previamente. Éstas incluyen `snmp-p` y `host-durer`. El primer filtro, llamado `snmp p`, buscará la cadena `p` en el encabezado de un paquete SNMP, el cual puede ayudar a rastrear cuáles sistemas están sondeando los dispositivos SNMP. El segundo filtro, `host durer`, presentará todo el tráfico enviado al *host* llamado `durer`. Como se puede observar, usando los otros botones, los filtros pueden modificarse, copiarse o borrarse muy fácil y rápidamente.

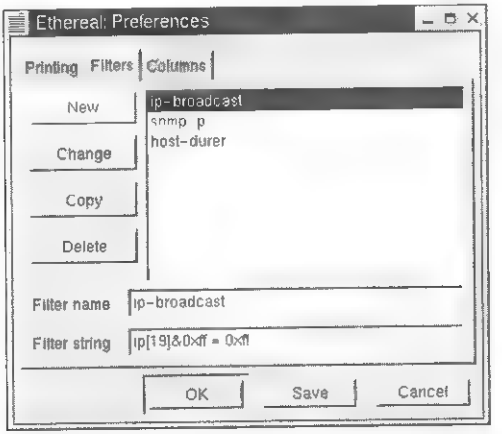


Figura 5.7 Selección del filtro de difusión IP.

Si se quiere usar un filtro mientras se captura el tráfico de red, se selecciona el filtro y luego se da clic en el botón `OK`. Suponiendo que se selecciona el filtro `ip broadcast`, **ethereal** pone la definición del filtro en el cuadro de texto de filtro de la ventana de captura, como se presenta en la figura 5.8.

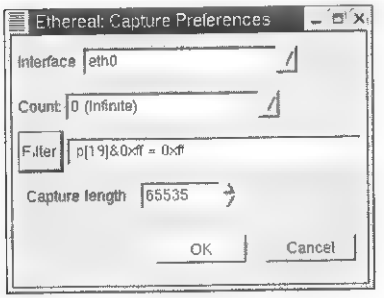


Figura 5.8 Filtro **ethereal** presentado.

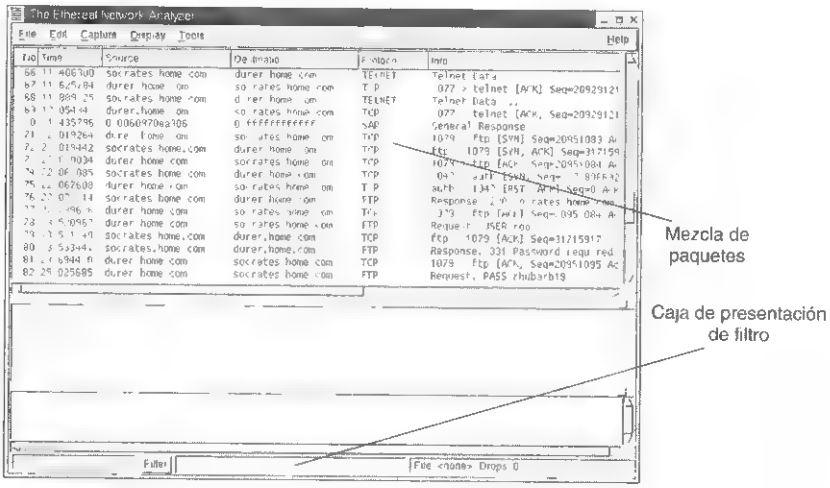


Figura 5.9 Lista extensa de paquetes clasificados por tipo de protocolo.

Si se selecciona el botón `OK` en esta ventana, se activará el filtro y empezará la captura del tráfico de red.

El filtro de presentación previamente mencionado también puede usarse para seleccionar un subconjunto de paquetes que ya hayan sido capturados. Por ejemplo, obsérvese que en la figura 5.9 se han obtenido paquetes `FTP`, `TCP` y `Telnet`.

¿Qué pasaría si se quisiera examinar sólo los paquetes `FTP`? En este caso, se entra la cadena de filtro `FTP` en el cuadro `Filter` en la ventana **ethereal** principal, seguida de la tecla *Enter*. La figura 5.10 muestra el resultado de esta acción. Como se observa,

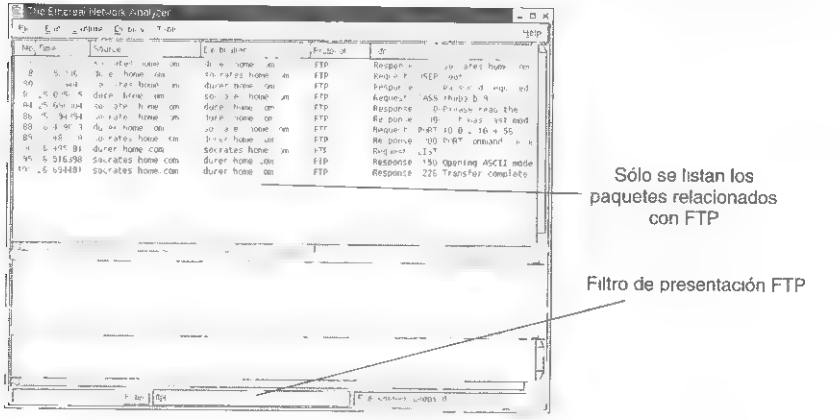


Figura 5.10 Filtro de presentación FTP.

el número de paquetes se ha reducido considerablemente. Lo que es interesante acerca de los filtros de presentación es que se pueden traer de vuelta los paquetes que fueron borrados del panel de resumen. En este caso, sólo se remueve el filtro `TCP` y se presiona la tecla Enter. Una vez hecho esto, se restablecerá la lista completa de paquetes. La selección de paquete es una poderosa característica de este software, y proporciona una muy buena forma de limitar la lista cuando se está interesado en revisar sólo unos cuantos paquetes específicos.

Téngase en cuenta que FTP usa el protocolo de transporte TCP y que los paquetes FTP adicionales pueden ser vistos como TCP. La razón de esto es que **ethereal** está clasificando los paquetes por el puerto que usa el protocolo. Como resultado, si se filtran sólo los paquetes FTP usando el filtro de presentación, se terminará obteniendo todos los paquetes que usaron el puerto FTP. Esto tiene el desafortunado efecto de no presentar otros paquetes que pueden contener información FTP.

Una característica útil de **ethereal** es que se pueden guardar los datos capturados en un archivo regular UNIX y, también, leer los archivos de captura creados por otras herramientas de software, como **tcpdump** o **snoop**. La herramienta **snoop** proporciona las mismas capacidades y características básicas que **tcpdump**, pero sólo está disponible para la plataforma Sun Solaris. En contraste, **tcpdump** está disponible para muchos sistemas UNIX.

Para guardar datos de una captura, sólo se selecciona el elemento `Save` del menú File (*Archivo*). Cuando se haya hecho esto, se presenta un cuadro de diálogo, como lo muestra la figura 5.11. Si se va a hacer una gran cantidad de capturas, se recomienda crear un directorio especial que contenenga todos los archivos. Además, puede ser necesario etiquetar los archivos con información importante acerca de los datos capturados para que, más tarde, puedan ser identificados fácilmente. Sin la etiqueta completa y apropiada de los archivos, la única manera de saber a ciencia cierta lo que cada archivo contiene es leyéndolo en el software **ethereal**, y revisándolo.

Cuando el cuadro de diálogo aparece por primera vez, contiene el directorio y cualesquiera archivos asociados en el lugar donde el programa **ethereal** fue arrancado. Se puede especificar explícitamente el archivo destino en el cuadro de texto `Selection:` (*Selección:*), o navegar usando el menú desplegable. En este ejemplo, la ruta y el nombre de archivo

```
/home/capture/ftp-traffic-12-10-99.data
```

se ingresaron en el cuadro de texto `Selection:`. Como se sugirió, el nombre de archivo contiene tanto el tipo de paquetes como la fecha en que se hizo la captura. En muchos casos, esto sería suficiente; sin embargo, en el futuro sería útil información adicional, como el nombre del sistema de donde se hizo la captura, la información adicional del protocolo, u otra información relacionada. Nótese también que se proporcionan tres botones en la ventana de diálogo, para conveniencia del usuario. Éstos son los botones `Create Dir` (Crear directorio), `Delete File` (Borrar archivo) y `Rename File` (Renombrar el archivo). Dando clic en el botón `Create Dir`, se presenta un pequeño cuadro de diálogo que solicita el nombre del directorio que se va a crear. Entrando el nombre del directorio y dando clic en el botón `Create`, se creará el

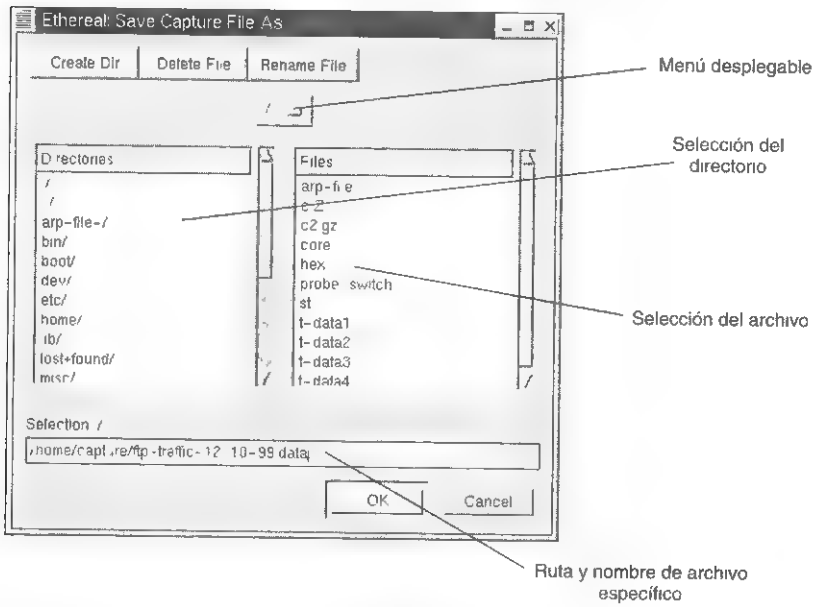


Figura 5.11 Ventana guardar **ethereal.**

directorio, a menos que haya ocurrido un error. Los dos botones restantes funcionan operando en los archivos seleccionados. Una vez seleccionado un archivo, cada botón llevará a cabo inmediatamente la acción requerida, tal como está descrito en la etiqueta del botón.

Establecimiento de preferencias y opciones

La herramienta **ethereal** proporciona algún control básico sobre la apariencia del panel de resumen de paquete principal, la definición

y mantenimiento de los filtros, y las opciones de impresión. Ya analizamos la operación de los filtros, ahora concentrémonos en las opciones de panel de resumen e impresión. Para tener acceso a las preferencias, se selecciona el elemento `Preferences...` del menú *Edit*. Se muestran tres marbetes que controlan el acceso a cada una de las pantallas asociadas. Seleccionando el marbete *Columns*, se presentarán las opciones del panel de resumen del paquete. La figura 5.12 muestra la ventana que se presentará cuando se selecciona este marbete.

El panel de resumen de la ventana principal presenta de manera predeterminada el número de paquete, sello de tiempo, dirección fuente, dirección destino, protocolo e información adicional del paquete. Una característica especialmente útil de **ethereal** es que estos campos son personalizables. El usuario puede controlar qué campos se presentan en el panel, y en qué orden. Es muy útil si el usuario necesita ver la información del paquete en uno o más formatos especializados. Una de las quejas continuas acerca de algunas de las herramientas de seguimiento existentes es la imposibilidad de producir reportes; con estas características personalizables, generar reportes porsonalizados es muy fácil.

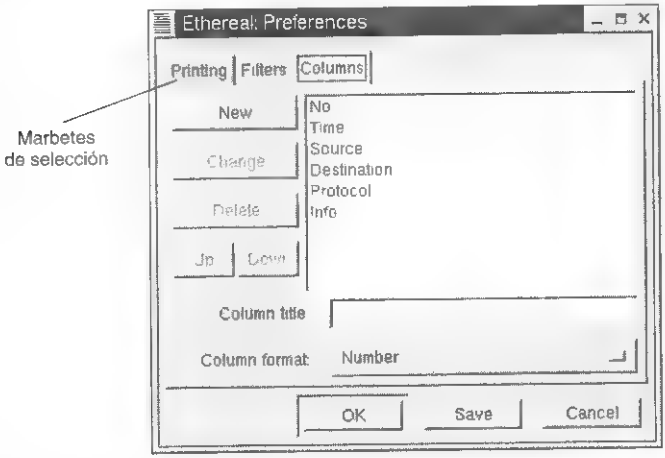


Figura 5.12 Ventana de preferencias **ethereal.**

Para tener acceso a las opciones de impresión, se da clic en el marbete *Printing* en el cuadro de diálogo de Preferencias **ethereal**. La figura 5.13 muestra el cuadro de diálogo que se presentará como resultado.

La herramienta **ethereal** puede imprimir paquetes en una impresora o en un archivo UNIX regular. El usuario puede definir una impresora y/o archivo predeterminados, para usar en las sesiones subsiguientes. En la figura 5.13, el comando UNIX **a2ps** se ha agregado al cuadro de texto `Command:` Este comando convierte la información ASCII (texto regular) al formato de lenguaje Postscript, para salida a una impresora compatible con Postscript. El comando tiene una gran cantidad de opciones que pueden controlar la salida, pero está predeterminado para presentar datos en dos páginas una junto a la otra en un modo panorámico (horizontal). Este formato, junto con un tamaño de fuente más pequeño, significa que se puede obtener una gran cantidad de datos en un número relativamente pequeño de páginas. Como se muestra en la figura 5.13, se definió un archivo de texto predeterminado, `ethereal.out`. Sin embargo, con la versión actual de software, este archivo no se usa cuando se imprime a un archivo. En su lugar, el usuario debe ingresar otro nombre de archivo cuando imprima a un archivo. La figura 5.14 muestra una toma instantánea de la pantalla de la salida **a2ps**, como se ve en una aplicación de visualización de Postscript llamada **ghostview**. Esto es lo que se esperaba ver cuando el archivo fuera impreso. Como se puede observar, usar el programa **a2ps** es una forma excelente de sacar información de paquete.

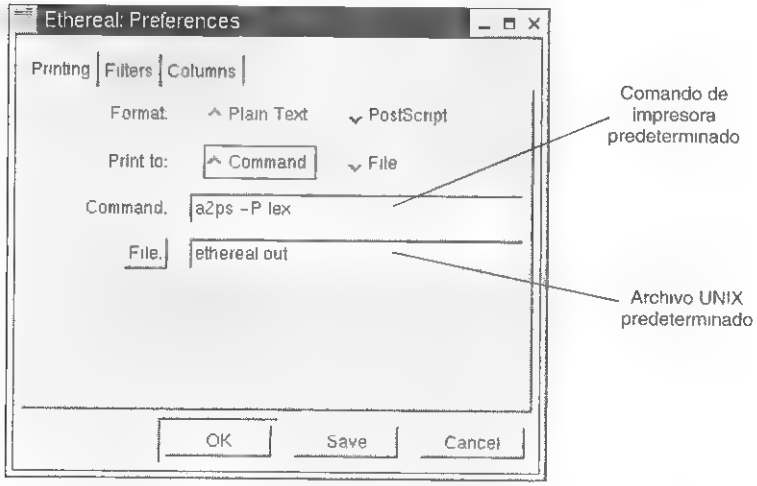


Figura 5.13 Preferencias de impresión **ethereal**.

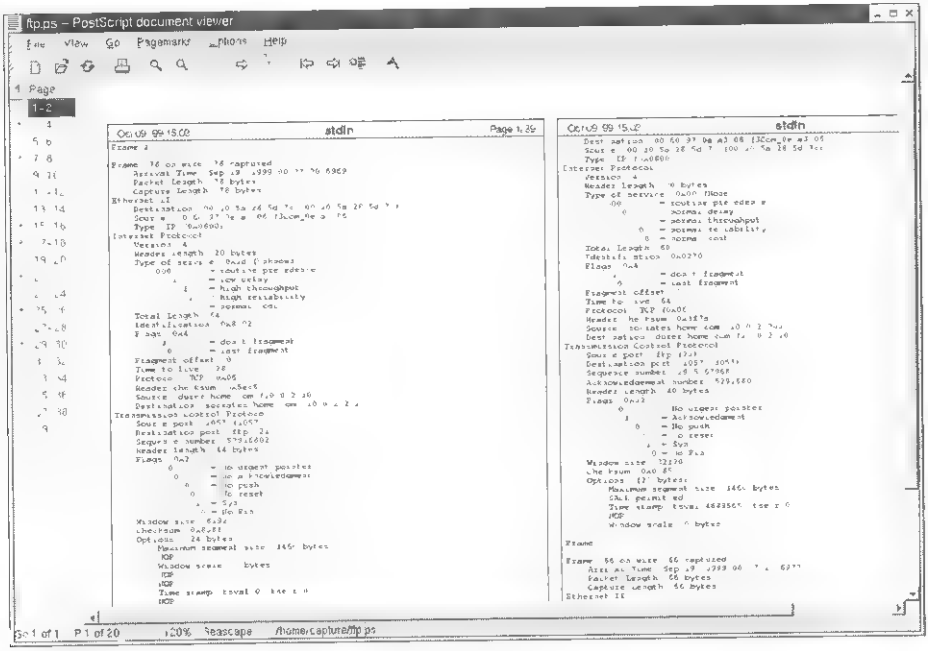


Figura 5.14 Avance de la salida **a2ps**.

Debido a que no todas las páginas son visibles en la figura 5.14, el usuario debe utilizar las barras de desplazamiento para examinar las porciones ocultas de la salida.

El software **ethereal** contiene una ventana de opciones de presentación que puede usarse para alterar el formato del campo `time` contenido en el panel de resumen. Si se selecciona el submenú Options... del menú `Display` presentará la ventana que se muestra en la figura 5.15.

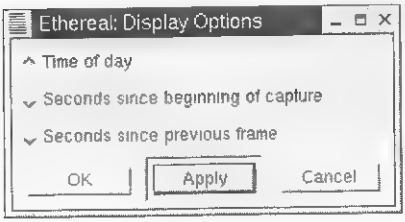


Figura 5.15 Ventana de opciones de presentación del campo time **ethereal.**

Una de las opciones de presentación más útiles es mostrar el tiempo en el que los paquetes llegaron al sistema. La opción `Time of day` presenta el momento exacto en que cada paquete fue leído por **ethereal.** Esto es muy útil en el rastreo de problemas de red que ocurrieron durante un periodo específico. La segunda opción presenta el tiempo como un desplazamiento (*offset*) desde cuando empezó la captura de paquetes. La opción final muestra el tiempo como un desplazamiento desde cada marco anterior. Como las opciones son mutuamente excluyentes, sólo se puede seleccionar una de ellas. El botón `Apply` (*Aplicar*) puede usarse para alterar los tiempos en el panel de resumen, sin descartar o borrar la ventana de opciones. Esto le permitirá al usuario ver cada opción de tiempo en el panel de resumen. La figura 5.16 muestra los paquetes de resumen, con la opción `Time of day` seleccionada.

Características adicionales

El menú `Tools` ofrece dos comandos de menú adicionales que la mayoría de los usuarios encontrará útiles. Primero, **ethereal** proporciona una forma de revisar todos los valores ASCII imprimibles y cierta información del protocolo que están contenidos en una conexión TCP. Usando el elemento de menú *Follow TCP Stream*, el usuario puede rastrear todos los datos que fluyeron entre los sistemas.

La figura 5.17 muestra el flujo TCP después de que un inicio de sesión FTP fuera capturado del sistema llamado `socrates`. En este ejemplo, `socrates` le permitió al usuario iniciar la sesión

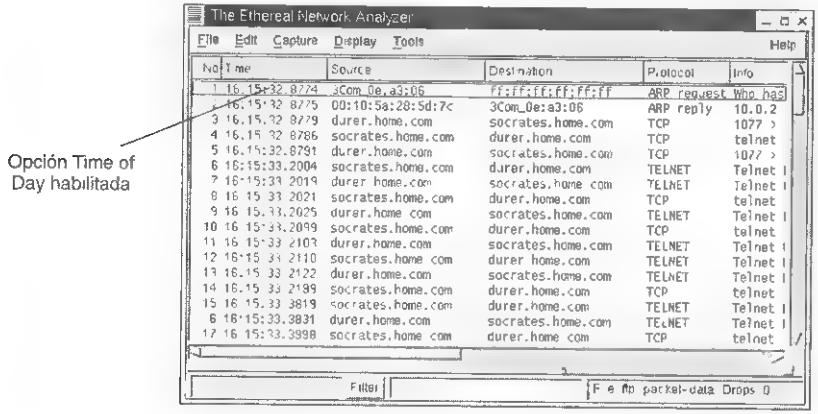


Figura 5.16 Paquetes mostrados con la opción Time of Day.

como `root`, cambiarse al directorio, y ejecutar un listado de directorio usando el comando FTP **dir**.

La ventana que se muestra en la figura 5.17 sólo contiene los datos de la secuencia del inicio de sesión, la transmisión de los comandos **CWD** y **LIST**, y la terminación de la información del saludo inicial. Se notará que no se presenta la información del listado del directorio. La razón para esto es simple: FTP usa una conexión TCP alterna para transferir la información del directorio. Como resultado, **ethereal** sólo muestra los datos de la primera conexión. Un aspecto interesante de usar la característica de flujo TCP es que ésta utiliza secretamente un filtro de presentación especial cuando presenta la información de la sesión TCP. Esto reduce los paquetes en el panel de resumen sólo a aquellos paquetes involucrados en la conexión dada. Para cancelar el efecto de mostrar una sola conexión TCP, se presiona la tecla `Enter` en el cuadro de texto de filtro de presentación que se muestra en la ventana principal **ethereal**.

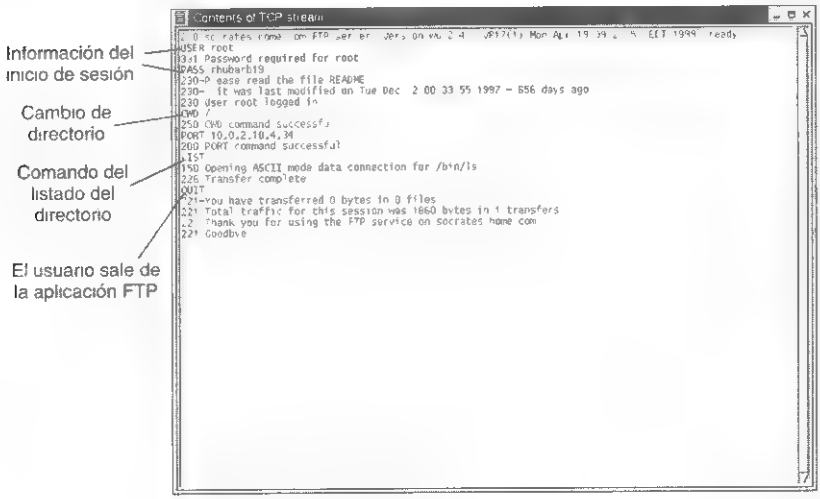


Figura 5.17 Información de inicio de sesión TCP.

Para examinar el listado de directorio FTP resultante del comando **LIST**, el usuario debe seleccionar un paquete de la segunda conexión. Para averiguar qué paquetes están asociados con una conexión TCP dada, el usuario debe examinar los puertos destino. Toda la información que fluye de una conexión FTP, como un listado de directorio o una transferencia de archivos,

usa el puerto 20. Todos los comandos FTP usan el puerto 21, el cual es el mismo puerto en el que la conexión arrancó. Una vez el usuario selecciona un paquete específico que es parte de la segunda conexión contenida en la ventana resumen, se presentará la información del directorio, como se muestra en la figura 5.18.

Como FTP usa una conexión TCP nueva para manejar cada transferencia de archivos o el listado del directorio, el usuario necesitará seleccionar los paquetes apropiados con aquellas conexiones para obtener los datos transferidos. La posibilidad de ver una transacción TCP completa usando este método es importante: le permite al usuario ver la operación interna de muchos de los protocolos y servicios TCP/IP. Esto puede tener un impacto profundo en la depuración y resolución de problemas de la red.

La otra opción del menú `Tools` proporciona un resumen de la información acerca de la captura. El submenú `Summary...` presenta tanto la información genérica como específica de los paquetes presentados en el panel de resumen de paquete. La figura 5.19 muestra la ventana de resumen **ethereal** de la sesión FTP que se acaba de analizar.

La ventana de resumen contiene tres secciones: `File`, `Data` y `Capture`. La sección `File` incluye el nombre del archivo (en el campo `Name`) donde se han guardado los paquetes. El campo `Length` muestra el tamaño en bytes de este archivo. El campo `Format` muestra qué formato de archivo se usó para almacenar los paquetes capturados. El formato `pcap` es el formato nativo de **ethereal**. El campo `Snapshot length` es el tamaño de paquete

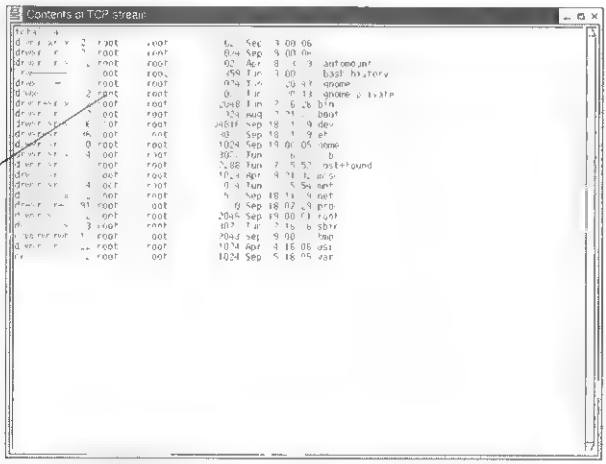


Figura 5.18 Listado de directorio FTP.

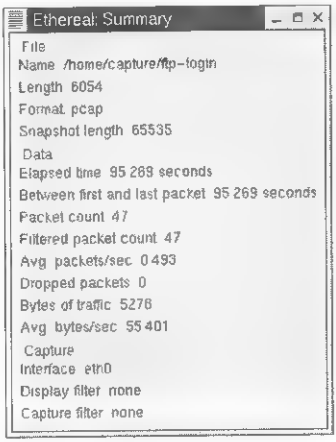


Figura 5.19 Ventana de resumen del paquete **ethereal.**

máximo que se usa cuando se captura tráfico de red y, como se mencionó previamente, es el valor predeterminado.

La sección `Data` incluye el campo `Elapsed Time` (*Tiempo transcurrido*), el cual indica la cantidad total de tiempo (en segundos) transcurrido desde la recepción del último paquete; por ejemplo, la duración de toda la sesión de captura del paquete. El siguiente campo, `Between first and last packet` (entre el primer y último paquete), es el mismo campo `Elapsed time`. El campo `Packet count` (*Conteo de paquete*) muestra la cantidad total de paquetes recibidos de la red o la cantidad contenida en un archivo UNIX. El campo `Filtered Packet count` (*Conteo de paquetes filtrados*) indica la cantidad total de paquetes que fueron aceptados por el filtro activo. En este caso, el filtro predeterminado aceptó todos los paquetes, y es por eso que contiene el mismo valor que el campo `Packet count`. El siguiente campo, `Avg. packets/sec` (*Promedio de paquetes por segundo*), contiene el número promedio de paquetes que entran a la interfaz de red en un intervalo de un segundo. El campo `Dropped packets` (*Paquetes abandonados*) indica la cantidad de paquetes que fueron abandonados durante la sesión de captura. La Figura 5.19 indica que no se abandonaron paquetes. Un número mayor de cero indica que la herramienta tuvo que abandonar paquetes, debido a que no se pudieron procesar suficientemente rápido.

El campo `Bytes of traffic` (*Bytes de tráfico*) representa qué cantidad de datos se transmitió en los paquetes, excluyendo cualquier porción de encabezado de los marcos de bajo nivel. El último campo, `Avg. bytes/sec` (*Promedio de bytes por segundo*),

muestra la tasa de trasmisión que mide los bytes por segundo durante el periodo de captura.

La sección `Capture` especifica el campo `Interface`, el cual muestra la interfaz en el sistema de donde fueron obtenidos los paquetes. En esta situación, como el paquete provino de un archivo UNIX, y no de una interfaz de red, el campo se marca como Unknown (*Desconocido*). El siguiente campo, `Display filter` (*Filtro de presentación*), muestra el nombre del filtro de presentación activo. En este caso, no está definido, y como resultado, se lista None (*Ninguno*). El último campo de esta sección, `Capture filter` (*Filtro de captura*), lista el filtro de captura activo. No se definió ningún filtro de captura activo: por tanto, en su lugar se presenta None.

El software **ethereal** suministra características adicionales que no se describen plenamente en este texto. Éstas incluyen:

- *match selected* (concordancia seleccionada)
- *colorize display* (presentación en colores)

La opción `Match Selected...` del menú Display explora cada paquete en busca de cierto patrón o criterio. Esto es muy útil si se está buscando una cadena o un valor específico después de capturar una gran cantidad de paquetes. En estos casos, la inspección manual no es exacta ni eficiente y, por tanto, se requiere una aproximación automatizada. La opción `Color Display...` del menú Display permite al usuario cambiar de colores el primer y segundo planos de ciertos tipos de paquete. Esto hace mucho más fácil la inspección y revisión del paquete.

Herramienta fping

Descripción general

El comando **fping** es muy similar al comando **ping** previamente analizado en este capítulo. Sin embargo, el software **fping** proporciona unas pocas funciones especiales que el **ping** normal no suministra. Como su contraparte, **fping** usa Solicitudes y respuestas Echo ICMP para determinar el alcance del sistema. Sin embargo, a diferencia del **ping** regular, se puede especificar una lista de nombres de *host* o direcciones IP en la línea de comandos. Igualmente, **fping** procesará un archivo que contenga una lista de nombres de *host* o direcciones IP. En lugar de enviar una solicitud a un objetivo único (como lo hace **ping**), **fping** enviará una solicitud ICMP a cada objetivo usando un mecanismo circular

(*round-robin*). Otra diferencia importante es que **fping** estaba destinado inicialmente a ser usado en la programación de secuencias de comandos, no como un comando de usuario como el **ping** normal. De todas formas, cuando se usa en la línea de comandos, **fping** puede abordar efectivamente algunos problemas comunes relacionados con las direcciones IP.

Las opciones de línea de comandos disponibles con **fping** pueden dividirse en dos secciones. La primera controla las diferentes opciones de presentación que pueden usarse para personalizar la salida. La segunda incluye parámetros operacionales que controlan la manera en que **fping** obtiene la información del alcance de los dispositivos, y procesa esa información.

De manera predeterminada, cuando un objetivo responde a una solicitud ICMP, éste es percibido y removido de la lista de objetivos que deberían ser revisados. Si un objetivo no responde en el tiempo establecido o límite de reintento, se considera inalcanzable. El comando **fping** puede ser usado para hacer ciclos indefinidamente (como el **ping** normal) o enviar un número de solicitudes ICMP definido por el usuario, a un objetivo.

Opciones de presentación

La tabla 5.5 incluye las diversas opciones de presentación disponibles con **fping**. Algunas de estas opciones son evidentes por sí mismas y no necesitan explicación adicional a la descripción asociada con cada opción en particular. Otras requieren una explicación más detallada y se describen más adelante.

Tabla 5.5 Lista de opciones de presentación **fping**.

Opción	Descripción
`-A`	Muestra las direcciones de los objetivos, en lugar de los nombres de *host*
`-Q`	Similar a la opción `-q`, pero muestra la información del resumen de resultados cada `n` segundos. Desafortunadamente, este comando parece que no funciona
`-a`	Muestra los sistemas que están activos y alcanzables, dada una lista de objetivos en la línea de comandos

Tabla 5.5 Lista de opciones de presentación **fping**. (*continuación*)

Opción	Descripción
`-d`	Muestra los nombres *host* de los objetivos, en lugar de las direcciones. Ésta es lo contrario de la opción `-A`
`-e`	Muestra los tiempos de ida y vuelta desde cuando el paquete se envió hasta recibir una respuesta
`-q`	Modo silencioso. No presenta resultados, pero establece el estado de retorno en la salida
`-u`	Muestra los sistemas que no están activos, dada una lista de objetivos en la línea de comandos
`-v`	Presenta la información de la versión del comando y la dirección de correo electrónico para enviar comentarios y reportes de *bugs*

Una de las características especialmente útiles de **fping** es que, con la obción `-a` seguida de una lista de objetivos, presentará los sistemas que se puede alcanzar. Por ejemplo, considérese el siguiente comando:

```
# fping -a vermeer monet rodin louvre
monet
rodin
louvre
```

Se presentan los *hosts* `monet` `rodin` y `louvre`, indicando que están activos, mientras `vermeer` no fue presentado, ya que no respondió.

Por el contrario, la lista opuesta se puede presentar con la opción `-u`. En este caso, los *hosts* que no responden son presentados así:

```
# fping -u vermeer monet rodin louvre
vermeer
```

Podemos ir un paso más allá y poner una lista de objetivos dentro de un archivo para su procesamiento. Supongamos que un archivo llamado `host list` contiene las siguientes direcciones IP:

```
10.0.2.75
10.0.2.100
```

```
10.0.2.150
10.0.2.151
10.0.2.152
10.0.2.153
10.0.2.200
10.0.2.201
```

Redirigiendo la entrada al archivo `host list`, **fping** puede procesar una gran cantidad de objetivos, en lugar de requerir que se especifiquen uno por uno en la línea de comandos:

```
# fping < host-list
10.0.2.75 is alive
10.0.2.100 is alive
10.0.2.150 is alive
10.0.2.151 is alive
10.0.2.152 is alive
10.0.2.153 is alive
10.0.2.200 is alive
10.0.2.201 is alive
```

Ésta es una forma más razonable de manejar una gran cantidad de *hosts*.

Si se prefiere, la opción `-f` también puede usarse en lugar de la redirección de archivo anterior. Así, los comandos

```
# fping < host-list
# fping -f host-list
```

son funcionalmente equivalentes y arrojan el mismo resultado. Sin embargo, la opción `-f` está restringida al usuario `root`, mientras cualquier usuario del sistema puede usar la opción de redireccionamiento que se mostró en el primer ejemplo.

Las opciones activo (`-a`) y no activo (`-u`) son muy útiles y pueden usarse en una variedad de situaciones. Por ejemplo, supongamos que se quiere ser notificado por medio del correo electrónico cuando ciertos sistemas están abajo. Una secuencia de comandos simple, como se muestra en el listado 5.1, podría lograr esto.

Listado 5.1 Secuencia de comandos de ejemplo de **fping** para identificar los sistemas abajo.

```
1 #!/usr/bin ksh
2
3 #
4 # Simple script to notify user when
5 # systems are down
6 #
```

(*continúa*)

```
 7
 8 FILE=$1
 9
10 FPINGCMD=`fping -u < $FILE`
11 MAIL=/bin/mailx
12 USER=root
13
14 if [ "$FPINGCMD" != "" ]; then
15         $MAIL -s "down system(s)" -u $USER $FPINGCMD"
16 fi
```

Esta secuencia de comandos corre el comando **fping** (línea 10) hacia los *hosts* incluidos en el archivo especificado, y si cualesquiera dispositivos contenidos en el archivo no se pueden alcanzar, se le envía un correo al usuario `root`, como se muestra en la línea 15. Obviamente, podría escribirse una secuencia de comandos más robusta y elaborada, pero el punto aquí es que **fping** puede ayudar a automatizar algunas pequeñas tareas de administración de red.

En ocasiones especiales, podría ser necesario manipular directamente direcciones IP, en lugar de nombres de *hosts* como en los ejemplos anteriores. Cuando se ha dado una lista de nombres de *hosts*, sería bueno regresar las direcciones IP. La opción `A` le ordena a **fping** que presente la información de direcciones IP, en lugar del nombre de *host*. Por tanto, en nuestro ejemplo, en el listado 5.1, procesamos un archivo que contiene una lista de direcciones IP. En este caso, procesaremos una lista de nombres de *hosts* válidos. El archivo `hostname-list` contiene lo siguiente:

```
monet
durer
vermeer
rembrandt
lex-mark
gateway
rodin
louvre
```

Procesando este archivo con la opción `A`, obtenemos:

```
# fping -A < hostname-list
10.0.2.126 is alive
134.111.2.1 is alive
10.0.2.75 is alive
10.0.2.75 is alive
10.0.2.129 is alive
128.197.2.200 alive
10.0.2.200 is alive
10.0.2.101 is alive
```

El comando **fping** no sólo proporciona una forma de determinar si los sistemas están operando en la red, sino que también puede suministrar alguna información de rendimiento. En particular, están disponibles unas pocas opciones que presentan varias estadísticas en cuanto al rendimiento de la fuente del comando **fping** a cada objetivo. La opción `-e` mostrará el tiempo de ida y vuelta o el tiempo transcurrido desde el momento en que el paquete se envía y hasta que se recibe una respuesta. El comando

```
# fping -e rembrandt
rembrandt is alive (2.82 ms)
```

proporciona una salida muy similar a la que se obtiene cuando se usa el comando **ping**. Otra opción útil, `s`, muestra un perfil estadístico acumulativo más completo del objetivo:

```
# fping -s rembrandt
rembrandt is alive
     1 targets
     1 alive
     0 unreachable
     0 unknown addresses
     0 timeouts (waiting for response)
     1 ICMP Echos sent
     1 ICMP Echo Replies received
     0 other ICMP received
3.09 ms (min round trip time)
3.09 ms (avg round trip time)
3.09 ms (max round trip time)
     0.015 sec (elapsed real time)
```

Esta opción proporciona una clasificación detallada que muestra la cantidad total de objetivos y la cantidad de objetivos que se encontraron activos (en este caso, 1). La otra información se explica por sí sola, y el tiempo real transcurrido se mide en segundos. Para determinar la exactitud de esta información, considérese este ejemplo:

```
# date; fping -s rodin; date
Sat Oct 3 13:25:01 EDT 1998
rodin is unreachable
       1 targets
       0 alive
       1 unreachable
       0 unknown addresses
       4 timeouts (waiting for response)
       4 ICMP Echos sent
```

(*continúa*)

```
      0 ICMP Echo Replies received
      0 other ICMP received
0.00 ms (min round trip time)
0.00 ms (avg round trip time)
0.00 ms (max round trip time)
      4.080 sec (elapsed real time)
Sat Oct 3 13:25:05 EDT 1998
```

Aquí, **fping** es rodeado por el comando **date** UNIX. El programa **date** se ejecuta antes de que el **fping** arranque y después de que termina su ejecución. Descontando cualesquiera retardos del sistema operativo, el tiempo transcurrido es bastante exacto. El tiempo transcurrido podría no ser 100% exacto, debido a varios factores, pero para muchas aplicaciones debería ser suficientemente exacto. Para este ejemplo fue escogido un sistema inalcanzable a fin de que **fping** fuera forzado a enviar paquetes adicionales y esperara la máxima cantidad de tiempo. Esto da una mayor cantidad de tiempo para medir las solicitudes, sin recurrir a opciones adicionales necesarias cuando el objetivo es un sistema activo.

Opciones operacionales

La tabla 5.6 incluye muchas opciones operacionales disponibles con **fping**. Algunas de estas opciones son evidentes por sí mismas y no necesitan explicación además de la descripción dada para cada opción en particular. Otras opciones requieren una explicación más en detalle y se describen a continuación.

Tabla 5.6 Lista de opciones operacionales **fping**.

Opción	Descripción
`-b`	Número de bytes de datos a incluir en la solicitud **ping**
`-B`	Esta opción, seguida de un valor, se utiliza para determinar el tiempo de retardo que **fping** emplea cuando está esperando respuestas. El valor se usa como un multiplicador y debe ser dado en un formato de punto flotante
`-c`	Determina el número de solicitudes enviadas a cada objetivo
`-C`	Similar a `-c`, pero las estadísticas por objetivo son presentadas en un formato que es más fácil de analizar sintácticamente

Tabla 5.6 Lista de opciones operacionales **fping**. (*continuación*)

Opción	Descripción
`-f`	Obtiene la lista de objetivos del archivo especificado. Esto está restringido al usuario root. Los usuarios regulares deben usar la redirección de archivo, en lugar de esta opción
`-l`	Envía una cantidad indefinida de solicitudes al objetivo; puede ser detenido con **^c** (**control-c**)
`-p`	Establece la cantidad de tiempo en milisegundos que **fping** esperará entre paquetes sucesivos para un objetivo. Esto sólo es significativo con las opciones `-l`, `-c` o `-C`
`-r`	Número de intentos que hará **ping** a un objetivo especificado. El valor predeterminado es 3
`-t`	Cantidad de tiempo para esperar la primera respuesta a la solicitud inicial de paquete. Lo predeterminado es 500 milisegundos

El comando **fping** envía una cantidad de solicitudes a un objetivo, esperando más tiempo una respuesta por cada solicitud sucesiva. El valor empleado está determinado por un factor de espera, el cual usa un multiplicador para cada solicitud que no reciba una respuesta. El valor de `1.5` es el multiplicador predeterminado. Para cambiar este valor, se usa la opción `-B` seguida de un valor de punto flotante. Por ejemplo, si se usa el valor predeterminado, se observa lo siguiente:

```
# fping -s rembrandt
rembrandt is unreachable
       1 targets
       0 alive
       1 unreachable
       0 unknown addresses
       4 timeouts (waiting for response)
       4 ICMP Echos sent
       0 ICMP Echo Replies received
       0 other ICMP received
0.00 ms (min round trip time)
0.00 ms (avg round trip time)
0.00 ms (max round trip time)
       4.077 sec (elapsed real time)
```

Usando la opción `-B`, se puede duplicar el factor de retardo para cada paquete enviado. Como resultado, se ve una cantidad significativa de tiempo empleado para cada una de las solicitudes de paquetes:

```
# fping -B 3.0 -s rembrandt
rembrandt is unreachable
       1 targets
       0 alive
       1 unreachable
       0 unknown addresses
       4 timeouts (waiting for response)
       4 ICMP Echos sent
       0 ICMP Echo Replies received
       0 other ICMP received
0.00 ms (min round trip time)
0.00 ms (avg round trip time)
0.00 ms (max round trip time)
      20.037 sec (elapsed real time)
```

Nótese que el tiempo transcurrido ha aumentado en aproximadamente `20` segundos, ya que el factor de retraso es multiplicado por cada paquete enviado.

Para controlar el número de solicitudes que **fping** envía a cada objetivo, se usa la opción `-c`. Se hacen cuatro intentos de manera predeterminada antes de renunciar al sistema especificado. La opción `-c` puede usarse para cambiar el número de intentos:

```
# fping -c 5 monet
monet : [0], 84 bytes, 3.55 ms (3.55 avg, 0% loss)
monet : [1], 84 bytes, 2.53 ms (3.04 avg, 0% loss)
monet : [2], 84 bytes, 1.63 ms (2.57 avg, 0% loss)
monet : [3], 84 bytes, 1.99 ms (2.42 avg, 0% loss)
monet : [4], 84 bytes, 1.64 ms (2.26 avg, 0% loss)
monet : xmt/rcv/%loss = 5/5/0%, min/avg/max = 1.63/2.26/3.55
```

Herramienta nmap

La herramienta **nmap** es una utilidad que localiza e identifica todos los puertos TCP y UDP disponibles en un sistema *host* objetivo. Conocido también como un explorador de puerto, **nmap** explorará los puertos TCP de `0` a `1024` (por defecto), para determinar el servicio escuchando los puertos que acepten una conexión. El software está construido para maximizar el ancho de banda de la red mientras se mantienen los recursos de procesamiento en el *host* usado para correr el **nmap**.

Tradicionalmente, **nmap** es visto más como una herramienta para hacer seguimientos de seguridad; se emplea con el fin de localizar las debilidades de seguridad en un sistema, en lugar de hacerlo como una herramienta de administración de red. Sin embargo, los administradores de red encontrarán muy útil esta herramienta para identificar los servicios de red en sistemas desconocidos. Se debe tener precaución en lo siguiente: la herramienta ha sido utilizada para ayudar a la gente a obtener acceso ilegal a las redes y sistemas asociados en toda la Internet. Por esto, **nmap** ha recibido alguna publicidad negativa. Si el usuario emplea esta herramienta en sus propias redes, puede estar tranquilo, pero si intenta usar esta herramienta en redes, sistemas o sitios que no controla, debería conseguir primero un permiso. Debe tenerse presente que: tener respeto por la privacidad de las redes y los sistemas de otros significa que el favor puede devolvérsele algún día.

En ocasiones, se pueden agregar a las redes dispositivos que interfieren con las operaciones normales de red. Además, el origen o la naturaleza de estos sistemas pueden no ser evidentes y el agente SNMP puede estar inactivo, debido a que puede estar inhabilitado o no configurado por alguna razón. Puede volverse necesario explorar los dispositivos para aprender más acerca de los servicios que ellos proporcionan. Por ejemplo, supóngase que un dispositivo adicionado a la red se considera la causa de un problema en el rendimiento de ésta, pero debido a que el dispositivo no parece soportar servicios como **telnet**, es difícil identificarlo.

Se puede hacer un **ping** al dispositivo, pero éste no suministra realmente mucha información:

```
# ping 10.0.2.10
PING 10.0.2.10 (10.0.2.10): 56 data bytes
64 bytes from 10.0.2.10: icmp_seq=0 ttl=128 time=0.7 ms
64 bytes from 10.0.2.10: icmp_seq=1 ttl=128 time=0.7 ms

--- 10.0.2.10 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 0.7/0.7/0.7 ms
```

Se puede también hacer un **telnet** al dispositivo, pero la conexión nunca se hará porque el dispositivo rechaza este tipo de acceso:

```
# telnet 10.0.2.10
Trying 10.0.2.10...
telnet: Unable to connect to remote host: Connection
refused
```

Sin embargo, se puede dejar que **nmap** sondee el dispositivo y ayude a determinar algo acerca de él.

```
nmap -O 10.0.2.10
```

A medida que se va realizando el proceso, la salida del comando **nmap** encaja con el perfil de la estación de trabajo de Windows 95:

```
Starting nmap V. 2.12 by Fyodor (fyodor@dhp.com, www.insecure.org/nmap/)
Interesting ports on (10.0.2.10):
Port       State      Protocol   Service
135        open       tcp        loc-srv
139        open       tcp        netbios-ssn
427        open       tcp        svrloc

TCP Sequence Prediction:Class=trivial time dependency
                        Difficulty=13 (Easy)
Remote operating system guess: Windows NT4 / Win95 / Win98
```

Normalmente, no sería fácil determinar que un dispositivo problemático fuera una estación de trabajo de Windows, pero usando **nmap**, muchos sistemas pueden identificarse muy rápida y fácilmente.

La herramienta **nmap** puede también examinar los servicios de trabajo en red corriendo en sistemas conocidos en la red. Esto es útil al determinar cuáles servicios del sistema están disponibles a través de varios sistemas. Por ejemplo, la siguiente salida se produjo usando **nmap** en un sistema de estación de trabajo Linux llamado `socrates`:

```
# nmap -O socrates

Starting nmap V. 2.12 by Fyodor (fyodor@dhp.com,
www.insecure.org/nmap/)
Interesting ports on socrates.nome.com (10.0.2.201):
Port       State      Protocol   Service
21         open       tcp        ftp
22         open       tcp        ssh
23         open       tcp        telnet
25         open       tcp        smtp
80         open       tcp        http
111        open       tcp        sunrpc
512        open       tcp        exec
513        open       tcp        login
514        open       tcp        shell
515        open       tcp        printer
610        open       tcp        npmp-local
```

(*continúa*)

```
615     open      tcp         unknown
620     open      tcp         unknown
982     open      tcp         unknown
1013    open      tcp         unknown
1024    open      tcp         unknown
6000    open      tcp         X11

TCP Sequence Prediction: Class=random positive increments
                         Difficulty=5369390 (Good luck!)
Remote operating system guess: Linux 2.1.122 - 2.1.132;
2.2.0-pre1 - 2.2.2
```

La opción `O` empleada aquí le dice a **nmap** que nos gustaría que intentara identificar el sistema objetivo usando lo que él llama *huellas digitales* TCP/IP. Básicamente, significa que se usan varias técnicas para detectar diferencias en los sistemas operativos o en el software que está corriendo en el dispositivo objetivo. Usando los datos obtenidos del dispositivo, **nmap** compara esta información con las huellas digitales de los dispositivos y sistemas conocidos. Cada dispositivo que **nmap** intenta identificar, se compara con los datos almacenados en una archivo llamado `nmap os-fingerprints`, el cual está localizado en el directorio `/usr/local/lib/nmap`. Este archivo contiene un gran número de huellas digitales para sistemas y dispositivos de trabajo en red. Si se encuentra un dispositivo que no concuerda con las entradas en el archivo de huellas digitales, se puede considerar el envío de la exploración al autor de **nmap**.

El software **nmap** soporta varias funciones importantes y varias opciones de línea de comandos. Además, proporciona una gran cantidad de técnicas de exploración del sistema, como se muestra en la tabla 5.7. Esta tabla también incluye la opción de comando asociada y una descripción general del proceso que se desarrolla con cada método de exploración.

La herramienta **nmap** fue diseñada para trabajar con mucha eficiencia cuando se sondea una gran cantidad de *hosts* en muchas redes diferentes. Además, es considerado uno de los exploradores de puerto más rápidos disponibles de dominio público e, igualmente, incluso un rival de productos comerciales. Las opciones de línea de comandos están divididas en dos categorías: la primera es opciones de exploración, listadas en la tabla 5.7; la segunda es opciones adicionales, la cual refina o controla el comportamiento de la herramienta. La tabla 5.8 presenta estas opciones adicionales.

Tabla 5.7 Técnicas de exploración **nmap**.

Método de Exploración	Opción de línea de comandos	Descripción
Puerto UDP	`-sU`	Explora los puertos UDP disponibles en el rango de `1` a `1024`, más los puertos listados en el archivo `/usr/local/lib/nmap/nmap-services`
TCP connect	`-sT`	Explora los puertos TCP disponibles en el rango de `1` a `1024`, más los puertos listados en el archivo `/usr/local/lib/nmap/nmap-services`; usa la llamada del sistema `connect` de bajo nivel, en un intento por establecer conexión con el sistema objetivo
TCPSYN (media)	`-sS`	Explora los puertos TCP usando la técnica *medio abierta*, lo que significa que sólo un lado de una conexión TCP está abierta y esperando reconocimiento
FTP Proxy	`-P`	Explora la devolución de FTP; usa la característica *proxy* en los servicios FTP para intentar la conexión
Reverse Indent (Sangría contraria)	`-I`	Habilita la exploración de sangría contraria TCP, la cual exige que el servidor `ident` esté corriendo; cuando se habilita, permite la identificación del propietario del proceso que usa un puerto TCP

Tabla 5.7 Técnicas de exploración **nmap**. (*continuación*)

Método de Exploración	Opción de línea de comandos	Descripción
Ping Sweep	`-sP`	Usa ICMP cuando se exploran sistemas de sondeo. Esto es equivalente a emitir una solicitud **ping**
FIN	`-sF`	Conocida como FIN furtivo, esta opción de exploración usa el paquete FIN TCP, en un intento por provocar una repuesta
Xmax Tree	`-sX`	Similar a FIN, pero enciende los bits de FIN, URG y PUSH en el paquete TCP
Null Scan (Exploración nula)	`-sN`	Similar a FIN, pero apaga todos los indicadores TCP

Tabla 5.8 Opciones **nmap** adicionales.

Opción	Descripción
`-F`	Modo de exploración rápido; sólo explora los puertos encontrados en el archivo `nmap services`
`-i`	Obtiene la información del objetivo desde un archivo especificado, en la línea de comandos
`-m`	Presenta la salida mínima, la cual incluye el nombre del *host*, números de puerto, etc.
`-o`	Registra los resultados del programa en un archivo especificado
`-p`	Usa el puerto o rango especificado, en lugar de los predeterminados
`-v`	Habilita el modo expresivo, el cual proporciona información adicional en la actividad del programa

Como **nmap** soporta una gran cantidad de argumentos y opciones de línea de comandos, se usa la opción -h para listar un resumen:

```
# nmap -h
nmap V. 2.12 usage: nmap [Scan Type(s)] [Options] <host or net #1...[#N]>
Scan types
    -sT tcp connect() port scan
     sS tcp SYN stealth port scan (must be root)
    -sF,-sX,-sN Stealth FIN, Xmas, or Null scan (only works against UNIX).
    -sP ping "scan". Find which hosts on specified network(s) are up but
          don't port scan them
    -sU UDP port scan, must be r00t
    -b <ftp_relay_host> ftp "bounce attack" port scan
Options (none are required, most can be combined):
    -f use tiny fragmented packets for SYN, FIN, Xmas, or NULL scan
    -P0 Don't ping hosts (needed to scan www.microsoft.com and others)
     PT Use "TCP Ping" to see what nosts are up (for normal and ping scans)
    -PI21 Use "TCP Ping" scan with probe destination port of 21 (or
          whatever)
    -PI Use ICMP ping packet to determines hosts that are up
    -PB Do BOTH TCP & ICMP scans in parallel (TCP dest port can be specified
          after the 'B')
    -PS Use TCP SYN sweep rather than the default ACK sweep used in
          "TCP ping"
    -O Use TCP/IP fingerprinting to guess what OS the remote host is
          running
    -p <range> ports: ex: '-p 23' will only try port 23 of the nost(s)
          '-p 20 30,63000-' scans 20-30 and 63000-65535. default: 1-1024
+ /etc/services
    -Ddecoy_host1,decoy2,ME,decoy3[,...] Launch scans from decoy host(s)
          along with the real one. If you care about the order your
            real IP
          appears, stick "ME" somewhere in the list. Even if the target
           detects the scan, they are unlikely to know which IP is
           scanning them and which are decoys.
    -F fast scan. Only scans ports in /etc/services, a la strobe(1).
    -I Get identd (rfc 1413) info on listening TCP processes.
    -n Don't DNS resolve anything unless we have to (makes ping scans
          faster)
     R Try to resolve all hosts, even down ones (can take a lot of
          time)
    -o <logfile> Output scan logs to <logfile> in human readable.
     m <logfile> Output scan logs to <logfile> in machine parseable
          format.
     i <inputfile> Grab IP numbers or hostnames from file. Use ' ' for
          stdin
```

(continúa)

```
    -g <portnumber> Sets the source port used for scans. 20 and 53 are
         good choices.
    -S <your IP> If you want to specify the source address of SYN or FYN
         scan.
     v Verbose. Its use is recommended. Use twice for greater effect.
     h Help, print this junk. Also see http://www.insecure.org/nmap/
     V Print version number and exit.
     e <devicename>. Send packets on interface <devicename>
         (eth0,ppp0,etc.).
     q quash argv to something benign, currently set to "pine".
         (deprecated)
Hostnames specified as internet hostname or IP address. Optional '/mask'
    specifies subnet. For example: cert.org/24 or 192.88.209.5/24 or
         192.88.209.0-255 or '128.88.209.*' all scan CERT's Class C.
SEE THE MAN PAGE FOR MORE THOROUGH EXPLANATIONS AND EXAMPLES
```

Una de las fortalezas de **nmap** es que soporta muchas metodologías de exploración. Algunos exploradores sólo soportan la exploración TCP, la cual es muy útil pero tiene limitaciones. Por ejemplo, algunos dispositivos de trabajo en red no proporcionan servicios de trabajo en red TCP genéricos como lo hacen los sistemas UNIX tradicionales; ellos pueden soportar sólo un conjunto limitado de servicios UDP. En este caso, sería inútil el uso de un explorador de puerto que sólo soporta TCP. Considérese la siguiente exploración y salida asociada:

```
# nmap -sT probe.home.com

Starting nmap V. 2.12 by Fyodor (fyodor@dhp.com,
www.insecure.org/nmap/)
Interesting ports on probe.home.com (10.0.2.50):
(Not showing ports in state: filtered)
Port    State       Protocol    Service

Nmap run completed -- 1 IP address (1 host up) scanned in
66 seconds
```

Como se puede observar en esta salida, **nmap** no detectó ningún servicio en el sistema objetivo. Sin embargo, al explorar el uso de la opción `-sU` se instruye a **nmap** para que explore un rango de puertos UDP, en lugar del rango de puerto TCP predeterminado. Así, de manera bastante interesante,

```
# nmap -sU 10.0.2.50
```

genera la siguiente salida cuando se ejecuta:

```
WARNING: -sU is now UDP scan -  for TCP FIN scan use -sF

Starting nmap V. 2.12 by Fyodor (fyodor@dhp.com,
www.insecure.org/nmap/)
Interesting ports on probe.home.com (10.0.2.50):
Port       State      Protocol   Service
161        open       udp        snmp

Nmap run completed    1 IP address (1 host up) scanned in
12 seconds
```

Se encuentra un proceso SNMP escuchando el puerto estándar 161. Esto indica que este dispositivo sólo soporta SNMP y nada más. Si se quisiera, se podría consultar al dispositivo un poco más usando las herramientas basadas en SNMP, para determinar más información acerca del agente que reside en el dispositivo. *Véase* el capítulo 8, "Herramientas SNMP", para información adicional acerca de las herramientas usadas para cumplir esta tarea.

El comportamiento predeterminado de **nmap** consiste en usar el método de exploración del puerto TCP en los puertos TCP estándar que han sido incluidos en el archivo de servicios asociado. Normalmente, el archivo `nmap_services` está localizado en el directorio `/usr/local/lib/nmap`, y **nmap** usará estos puertos además de explorar todos los puertos en el rango de 1 a 1024. Usando sólo los valores predeterminados, **nmap** puede ser muy útil. Considérese el siguiente comando:

```
# nmap sparky
```

Éste muestra una gran cantidad de información en el *host* conocido como `sparky`:

```
Starting nmap V. 2.12 by Fyodor (fyodor@dhp.com, www.insecure.
org/nmap/)

Interesting ports on sparky.home.com (10.0.2.12):
Port  State     Protocol   Service
21    open      tcp        ftp
23    open      tcp        telnet
25    open      tcp        smtp
53    open      tcp        domain
79    open      tcp        finger
80    open      tcp        http
98    open      tcp        linuxconf
111   open      tcp        sunrpc
113   open      tcp        auth
119   open      tcp        nntp
513   open      tcp        login
```

(*continúa*)

```
514    open      tcp         shell
515    open      tcp         printer
652    open      tcp         unknown
673    open      tcp         unknown
678    open      tcp         unknown
683    open      tcp         unknown
713    open      tcp         unknown
1024   open      tcp         unknown
6000   open      tcp         X11

Nmap run completed -1 IP address (1 host up) scanned in 1 second
```

La salida anterior se obtuvo a partir de una exploración de un sistema Red Hat Linux 6.0. Como se puede observar, esta exploración muestra que están corriendo muchos de los servicios UNIX estándar. Sólo los servicios TCP son listados, ya que éste es el modo predeterminado; esto es equivalente a usar la opción `sT` de línea de comandos. Los servicios que corren en el sistema incluyen `ftp`, `Telnet`, `smtp` y otros. Igualmente, varios puertos no fueron identificados, y fueron marcados como Unknown (Desconocidos). Estos puertos incluyen servicios como `amd` en el puerto 713, el cual es un servicio de automontaje, y el puerto 678, el cual es un servicio `rpc rpc.mountd`, pero **nmap** no los reconoció como tales. Se pueden detectar servicios adicionales usando el comando **netstat** mientras se encuentre conectado al sistema. Este comando se describió en el capítulo anterior.

Si se quisiera explorar puertos UDP y TCP al mismo tiempo, se necesitaría usar las opciones `sU` y `sT` juntas. Si no se combinan, **nmap** sólo explorará una o la otra. El comando

```
# nmap -sU -sT sparky
```

generará lo siguiente:

```
WARNING: -sU is now UDP scan -- for TCP FIN scan use -sF

Starting nmap V. 2.12 by Fyodor (fyodor@dhp.com,
www.insecure.org/nmap/)
Interesting ports on socrates.home.com (10.0.2.201):
Port  State    Protocol  Service
21    open     tcp       ftp
23    open     tcp       telnet
25    open     tcp       smtp
53    open     udp       domain
53    open     tcp       domain
79    open     tcp       finger
80    open     tcp       http
98    open     tcp       linuxconf
```

(*continúa*)

```
111   open     udp       sunrpc
111   open     tcp       sunrpc
113   open     tcp       auth
119   open     tcp       nntp
161   open     udp       snmp
177   open     udp       xdmcp
513   open     tcp       login
514   open     tcp       shell
515   open     tcp       printer
517   open     udp       talk
518   open     udp       ntalk
650   open     udp       bwnfs
652   open     tcp       unknown
661   open     udp       unknown
671   open     udp       unknown
673   open     tcp       unknown
676   open     udp       unknown
678   open     tcp       unknown
681   open     udp       unknown
683   open     tcp       unknown
713   open     tcp       unknown
714   open     udp       unknown
800   open     udp       mdbs_daemon
1022  open     udp       unknown
1023  open     udp       unknown
1024  open     udp       unknown
1024  open     tcp       unknown
1026  open     udp       unknown
2049  open     udp       nfs
6000  open     tcp       X11

Nmap run completed -- 1 IP address (1 host up) scanned in 5
seconds
```

Nótese que ahora se presenta ambos puertos, UDP y TCP, clasificados por número de puerto.

Como con cualquier buen explorador de puerto, es primordial la cantidad de explorar un puerto en particular. Una buena forma de determinar si todos los sistemas en una red tienen un conjunto estándar de servicios de red, o un servicio en particular, es explorar la red con un número de puerto específico en mente. Por ejemplo, supóngase que nos gustaría determinar si todos los sistemas en la red `10.0.2.0` soportan alguna clasificación del agente SNMP. El comando:

```
# nmap -p 161 -sU -o results 10.0.2.0/24
```

le dice a **nmap** que explore el puerto `161` (el puerto SNMP) en la red `10.0.2.0` usando UDP, y luego guarde la información de

salida en un archivo llamado `results`. Si se presenta este archivo, se encuentra lo siguiente:

```
# Log of: nmap -p 161 -sU -o result 10.0.2.0/24
No ports open for host durer.home.com (10.0.2.10)
No ports open for host king.home.com (10.0.2.21)
No ports open for host derry.home.com (10.0.2.25)
Interesting ports on probe.home.com (10.0.2.50):
Port        State       Protocol    Service
161         open        udp         snmp

Interesting ports on switch.home.com (10.0.2.60):
Port        State       Protocol    Service
161         open        udp         snmp
Interesting ports on tux.home.com (10.0.2.65 ):
Port        State       Protocol    Service
161         open        udp         snmp

Interesting ports on didymus gw2.nome.com (10.0.2.129):
Port        State       Protocol    Service
161         open        udp         snmp

Interesting ports on didymus gw3.home.com (10.0.2.130):
Port        State       Protocol    Service
161         open        udp         snmp

Host        (10.0.2.255) seems to be a subnet broadcast address
(returned 2 extra pin
gs).  Skipping host.
```

A medida que esto resulta, éste es un mecanismo muy razonable para usar en un servicio de inventario a gran escala. Cualquier servicio TCP o UDP puede ser consultado usando este enfoque.

Este ejemplo demuestra otra característica poderosa de esta herramienta. Concretamente, se pueden especificar los sistemas o las redes objetivo utilizando unas pocas notaciones diferentes. Primero, se puede especificar una dirección IP usando una lista o rangos para cada parte de la dirección. Así, se puede explorar una clase IP completa con el carácter "*". Por ejemplo, `128.197.*.*` permite la exploración de toda la red clase B. Otra forma de expresar esto es empleando el carácter raya. Así, `128.197.1 255.1-255` es funcionalmente el mismo que cuando se usa la sintaxis `128.197.*.*` Segundo, se puede usar la notación máscara, como se muestra en el ejemplo del comando **nmap** anterior. Concretamente, `128.197.0.0/16` enmascarará y es equivalente a los dos ejemplos usando los caracteres "*" o "-". Finalmente, se puede usar una secuencia numerada combinada con la sintaxis de rango. Así, `128.197.90.1, 2, 3, 4, 5, 100-105` explorará las siguientes direcciones: `128.197.90.1, 128.197.90.2, 128.197.90.3, 128.197.90.4` y `128.197.90.5`, al igual que las direcciones

128.197.90.100 hasta 128.197.90.105. Usando estos formatos IP mejora inmensamente la facilidad de explorar subredes o redes completas.

Cuando se exploran redes, a veces es útil saber exactamente lo que **nmap** está haciendo todo el tiempo. Por esta razón, se ha suministrado la opción -v. Téngase en cuenta que, como resultado, puede generarse una gran cantidad de salida; por tanto podría ser importante usar esta opción con precaución. En el ejemplo anterior se exploró una subred completa. Si se agrega la opción -v con este comando y se reduce el número de objetivos a la mitad usando un rango IP, el comando **nmap -v -p 161 10.0.2.1-127** presentaría lo siguiente:

```
No scantype specified, assuming vanilla tcp connect() scan.
  Use -sP if you reall
y don't want to portscan (and just want to see what hosts
  are up).

Starting nmap V. 2.12 by Fyodor (fyodor@dhp.com,
  www.insecure.org/nmap/)
Host (10.0.2.0) appears to be down, skipping it.
Host (10.0.2.1) appears to be down, skipping it.
Host (10.0.2.2) appears to be down, skipping it.
Host (10.0.2.3) appears to be down, skipping it.
Host (10.0.2.4) appears to be down, skipping it.
Host (10.0.2.5) appears to be down, skipping it.
Host (10.0.2.6) appears to be down, skipping it.
Host (10.0.2.7) appears to be down, skipping it.
Host (10.0.2.8) appears to be down, skipping it.
Host (10.0.2.9) appears to be down, skipping it.
Host durer.home.com (10.0.2.10) appears to be up ... good.
Initiating TCP connect() scan against durer.nome.com
   (10.0.2.10)
The TCP connect scan took 0 seconds to scan 1 ports.
No ports open for host durer.home.com (10.0.2.10)
Host (10.0.2.11) appears to be down, skipping it.
Host (10.0.2.12) appears to be down, skipping it.
Host (10.0.2.13) appears to be down, skipping it.
Host (10.0.2.14) appears to be down, skipping it.
Host (10.0.2.15) appears to be down, skipping it.
Host (10.0.2.16) appears to be down, skipping it.
Host (10.0.2.17) appears to be down, skipping it.
Host (10.0.2.18) appears to be down, skipping it.
Host (10.0.2.19) appears to be down, skipping it.
Host (10.0.2.20) appears to be down, skipping it.
Host (10.0.2.21) appears to be down, skipping it.
Host (10.0.2.22) appears to be down, skipping it.
Host (10.0.2.23) appears to be down, skipping it.
```

(*continúa*)

```
Host (10.0.2.24) appears to be down, skipping it.
Host (10.0.2.25) appears to be down, skipping it.
Host (10.0.2.26) appears to be down, skipping it.
Host (10.0.2.27) appears to be down, skipping it.
Host (10.0.2.28) appears to be down, skipping it.
Host (10.0.2.29) appears to be down, skipping it.
Host (10.0.2.30) appears to be down, skipping it.
Host (10.0.2.31) appears to be down, skipping it.
Host (10.0.2.32) appears to be down, skipping it.
Host (10.0.2.33) appears to be down, skipping it.
Host (10.0.2.34) appears to be down, skipping it.
Host (10.0.2.35) appears to be down, skipping it.
Host (10.0.2.36) appears to be down, skipping it.
Host (10.0.2.37) appears to be down, skipping it.
Host (10.0.2.38) appears to be down, skipping it.
Host (10.0.2.39) appears to be down, skipping it
Host (10.0.2.40) appears to be down, skipping it.
Host (10.0.2.41) appears to be down, skipping it.
Host (10.0.2.42) appears to be down, skipping it.
Host (10.0.2.43) appears to be down, skipping it.
Host (10.0.2.44) appears to be down, skipping it.
Host (10.0.2.45) appears to be down, skipping it.
Host (10.0.2.46) appears to be down, skipping it.
Host (10.0.2.47) appears to be down, skipping it.
Host (10.0.2.48) appears to be down, skipping it.
Host (10.0.2.49) appears to be down, skipping it.
Host probe.nome.com (10.0.2.50) appears to be up ... good.
Initiating FIN,NULL, UDP, or Xmas stealth scan against
  probe.home.com (10.0.2.50)
The UDP or stealth FIN/NULL/XMAS scan took 0 seconds to
  scan 1 ports.
Interesting ports on probe.home.com (10.0.2.50):
Port    State      Protocol      Service
161     open       udp           snmp
Host (10.0.2.51) appears to be down, skipping it.
Host (10.0.2.52) appears to be down, skipping it.
Host (10.0.2.53) appears to be down, skipping it.
Host (10.0.2.54) appears to be down, skipping it.
Host (10.0.2.55) appears to be down, skipping it.
Host (10.0.2.56) appears to be down, skipping it.
Host (10.0.2.57) appears to be down, skipping it.
Host (10.0.2.58) appears to be down, skipping it.
Host (10.0.2.59) appears to be down, skipping it.
Host switch.nome.com (10.0.2.60) appears to be up ... good.
IInitiating FIN,NULL, UDP, or Xmas stealth scan against
  switch.home.com (10.0.2.60)
The UDP or stealth FIN/NULL/XMAS scan took 0 seconds to
  scan 1 ports.
```

(*continúa*)

```
Interesting ports on switch.home.com (10.0.2.60):
Port    State       Protocol    Service
161     open        udp         snmp
Host (10.0.2.61) appears to be down, skipping it.
Host (10.0.2.62) appears to be down, skipping it.
Host (10.0.2.63) appears to be down, skipping it.
Host (10.0.2.64) appears to be down, skipping it.
Host tux.home.com (10.0.2.65) appears to be up ... good.
IInitiating FIN,NULL, UDP, or Xmas stealth scan against
 switch.home.com (10.0.2.65)
The UDP or stealth FIN/NULL/XMAS scan took 0 seconds to
 scan 1 ports.
Interesting ports on tux.nome.com (10.0.2.65):
Port    State       Protocol    Service
161     open        udp         snmp
No ports open for host switch.home.com (10.0.2.60)
Host (10.0.2.66) appears to be down, skipping it.
Host (10.0.2.67) appears to be down, skipping it.
Host (10.0.2.68) appears to be down, skipping it.
Host (10.0.2.69) appears to be down, skipping it.
Host (10.0.2.70) appears to be down, skipping it.
Host (10.0.2.71) appears to be down, skipping it.
Host (10.0.2.72) appears to be down, skipping it.
Host (10.0.2.73) appears to be down, skipping it.
Host (10.0.2.74) appears to be down, skipping it.
Host (10.0.2.75) appears to be down, skipping it.
Host (10.0.2.76) appears to be down, skipping it.
Host (10.0.2.77) appears to be down, skipping it.
Host (10.0.2.78) appears to be down, skipping it.
Host (10.0.2.79) appears to be down, skipping it.
Host (10.0.2.80) appears to be down, skipping it.
Host (10.0.2.81) appears to be down, skipping it.
Host (10.0.2.82) appears to be down, skipping it.
Host (10.0.2.83) appears to be down, skipping it.
Host (10.0.2.84) appears to be down, skipping it.
Host (10.0.2.85) appears to be down, skipping it.
Host (10.0.2.86) appears to be down, skipping it.
Host (10.0.2.87) appears to be down, skipping it.
Host (10.0.2.88) appears to be down, skipping it.
Host (10.0.2.89) appears to be down, skipping it.
Host (10.0.2.90) appears to be down, skipping it.
Host (10.0.2.91) appears to be down, skipping it.
Host (10.0.2.92) appears to be down, skipping it.
Host (10.0.2.93) appears to be down, skipping it.
Host (10.0.2.94) appears to be down, skipping it.
Host (10.0.2.95) appears to be down, skipping it.
Host (10.0.2.96) appears to be down, skipping it.
Host (10.0.2.97) appears to be down, skipping it.
Host (10.0.2.98) appears to be down, skipping it.
```

(*continúa*)

```
Host (10.0.2.99) appears to be down, skipping it.
Host (10.0.2.100) appears to be down, skipping it.
Host (10.0.2.101) appears to be down, skipping it.
Host (10.0.2.102) appears to be down, skipping it.
Host (10.0.2.103) appears to be down, skipping it.
Host (10.0.2.104) appears to be down, skipping it.
Host (10.0.2.105) appears to be down, skipping it.
Host (10.0.2.106) appears to be down, skipping it.
Host (10.0.2.107) appears to be down, skipping it.
Host (10.0.2.108) appears to be down, skipping it
Host (10.0.2.109) appears to be down, skipping it.
Host (10.0.2.110) appears to be down, skipping it.
Host (10.0.2.111) appears to be down, skipping it.
Host (10.0.2.112) appears to be down, skipping it.
Host (10.0.2.113) appears to be down, skipping it.
Host (10.0.2.114) appears to be down, skipping it.
Host (10.0.2.115) appears to be down, skipping it.
Host (10.0.2.116) appears to be down, skipping it.
Host (10.0.2.117) appears to be down, skipping it.
Host (10.0.2.118) appears to be down, skipping it.
Host (10.0.2.119) appears to be down, skipping it.
Host (10.0.2.120) appears to be down, skipping it.
Host (10.0.2.121) appears to be down, skipping it
Host (10.0.2.122) appears to be down, skipping it.
Host (10.0.2.123) appears to be down, skipping it.
Host (10.0.2.124) appears to be down, skipping it.
Host (10.0.2.125) appears to be down, skipping it.
Host (10.0.2.126) appears to be down, skipping it.
Host didymus gw2.home.com (10.0.2.127) appears to be up ...
 good.
Initiating FIN,NULL, UDP, or Xmas stealth scan against
 didymus-gw2.home.com (10.0.2.127)
The UDP or stealth FIN/NULL/XMAS scan took 0 seconds to
 scan 1 ports.
Interesting ports on didymus-gw2.home.com (10.0.2.127):
Port    State         Protocol  Service
161     open          udp       snmp

Nmap run completed -- 127 IP addresses (5 hosts up) scanned
 in 1 second
```

Como se puede observar, se produjo una gran cantidad de salida, a pesar de que sólo se exploró la mitad de la subred. Igualmente, **nmap** identificó cuatro dispositivos que soportan agentes SNMP, como se notó por las líneas que contienen la cadena: `Host switch.home.com (10.0.2.60) appears to be up...good`. Nótese que el punto más importante en relación con este ejemplo es que **nmap** es muy inteligente respecto a sondear sistemas no existentes o abajo (*down*); no gastó mucho

tiempo o recursos en sondear sistemas que no respondieron. Antes de explorar un dispositivo, **nmap** primero determina si éste se puede alcanzar en la red ejecutando un **ping** en él. Esto no sólo reduce la cantidad de tiempo requerido para ejecutar la exploración y disminuir los recursos del sistema que se necesitan, sino que, igualmente, ayuda a preservar el ancho de banda de red. Es importante no subestimar el impacto que la exploración puede tener en una red, y **nmap** hace un buen trabajo al reducir los requerimientos de red mientras sondea.

Listado de salida nmap

El listado que se presenta a continuación muestra algo de la salida **nmap** cuando se usa con una serie de sistemas de computadora y dispositivos de trabajo en red diferentes. Ciertamente su utilidad puede variar, pero esta herramienta es muy buena para detectar diferentes tipos de sistemas. Esta sección incluye tanto la salida como una descripción general de los sistemas y los servicios de trabajo en red soportados. Como la mayoría de los sistemas soportan muchos de los mismos servicios y procesos de trabajo en red, los más comunes se incluyen en la tabla 5.9.

Tabla 5.9 Listados estándar **nmap** de red.

Puerto	Nombre del servicio	Descripción
7	`echo`	Reproduce los caracteres enviados a este puerto; servicio proporcionado por el proceso `inetd`; se emplea principalmente para pruebas
8	`discard`	Descarta cualquier dato enviado a él; actúa como `/dev/null` para los servicios de trabajo en red y otras aplicaciones de trabajo en red; se utiliza principalmente para pruebas
13	`daytime`	Proporciona el tiempo en un formato legible para el ser humano; se emplea principalmente para pruebas

Tabla 5.9 Listados estándar **nmap** de red. (*continuación*)

Puerto	Nombre del servicio	Descripción
19	chargen	Generador de carácter; produce el conjunto de carácter ASCII; se usa principalmente para pruebas
21	ftp	Servidor de protocolo de transferencia de archivos estándar
23	telnet	Servidor Telnet; proporciona servicios de inicio de sesión remota
25	smtp	Protocolo simple de transferencia de correo; usualmente el servidor sendmail está escuchando este puerto
37	time	Proporciona tiempo legible para la máquina
53	domain	Proceso del servidor de los nombres del dominio
79	finger	Proceso del servidor *finger*
80	http	Proceso del servidor Web
111	sunrpc	Servicios Sun de llamadas a procedimientos remotos de Sun
199	smux	Agente maestro SNMP
382	hp-managed-node	Agente HP. Este proceso proporciona servicios de administración de red para los productos del administrador de red HP
512	exec	Servidor de ejecución remota con autenticación
513	login	Inicio remoto con autenticación
514	shell	Servidor shell remoto con autenticación
515	printer	Servidor de impresión remota

Tabla 5.9 Listados estándar **nmap** de red. (*continuación*)

Puerto	Nombre del servicio	Descripción
540	uucp	Servidor UUCP
4045	lockd	Demonio lock

Sistema AIX

Descripción

Este reporte fue generado de un sistema RISC 6000 corriendo AIX 4.2. Incluye algunos de los servicios de trabajo en red UNIX estándar, y también algunos específicos.

Resultados del reporte

```
Starting nmap V. 2.12 by Fyodor (fyodor@dhp.com, www.insecure.
org/nmap/)
Interesting ports on lexis.home.com (10.0.2.240):
Port       State      Protocol   Service
7          open       tcp        echo
9          open       tcp        discard
13         open       tcp        daytime
19         open       tcp        chargen
21         open       tcp        ftp
23         open       tcp        telnet
25         open       tcp        smtp
37         open       tcp        time
111        open       tcp        sunrpc
199        open       tcp        smux
382        open       tcp        hp-managed-node
512        open       tcp        exec
513        open       tcp        login
514        open       tcp        shell
515        open       tcp        printer
543        open       tcp        klogin
544        open       tcp        kshell
651        open       tcp        unknown
652        open       tcp        unknown
777        open       tcp        unknown
2401       open       tcp        cvspserver

TCP Sequence Prediction: Class=truly random
                         Difficulty 9999999 (Good luck!)
Remote operating system guess: AIX 4.2
Nmap run completed - 1 IP address (1 host up) scanned in 4 seconds
```

Sistema Solaris

Descripción

Este reporte identifica el sistema llamado `chips` como un sistema Solaris 2.6 o Solaris 2.7. En este caso, el sistema es realmente 2.6, pero como existen relativamente pocas diferencias entre 2.6 y 2.7, **nmap** indica ambos. Dos servicios interesantes y únicos detectados por **nmap** son `lotusnote` y `xaudio`. El primer proceso representa un servidor Lotus Notes Domino, y el otro es una aplicación de audio/video que proporciona soporte de multidifusión.

Resultados del reporte

```
Starting nmap V. 2.12 by Fyodor (fyodor@dhp.com, www.insecure.
org/nmap/)
Interesting ports on chips.home.com (10.0.2.199):
Port       State      Protocol   Service
7          open       tcp        echo
9          open       tcp        discard
13         open       tcp        daytime
19         open       tcp        chargen
21         open       tcp        ftp
23         open       tcp        telnet
25         open       tcp        smtp
37         open       tcp        time
79         open       tcp        finger
80         open       tcp        http
111        open       tcp        sunrpc
512        open       tcp        exec
513        open       tcp        login
514        open       tcp        shell
515        open       tcp        printer
540        open       tcp        uucp
1103       open       tcp        xaudio
1352       open       tcp        lotusnote
4045       open       tcp        lockd
6112       open       tcp        dtspc
7100       open       tcp        font service

TCP Sequence Prediction: Class=random positive increments
                         Difficulty=48025 (Worthy challenge)
Remote operating system guess: Solaris 2.6 - 2.7

Nmap run completed   1 IP address (1 host up) scanned in 2 seconds
```

Sistema HP-UX

Descripción

Este sistema, llamado `dip`, proporciona muchos de los servicios estándar e incluye Klogin/Kshell para inicio de sesión y ejecución de shell remoto seguros en los puertos 543 y 544, respectivamente.

Resultados del reporte

```
Starting nmap V. 2.12 by Fyodor (fyodor@dhp.com, www.inse-
cure.org/nmap/)
Interesting ports on dip.home.com (10.0.2.198):
Port    State   Protocol  Service
7       open    tcp       echo
9       open    tcp       discard
13      open    tcp       daytime
19      open    tcp       chargen
21      open    tcp       ftp
23      open    tcp       telnet
25      open    tcp       smtp
37      open    tcp       time
53      open    tcp       domain
94      open    tcp       objcall
111     open    tcp       sunrpc
113     open    tcp       auth
135     open    tcp       loc srv
382     open    tcp       hp-managed-node
512     open    tcp       exec
513     open    tcp       login
514     open    tcp       shell
515     open    tcp       printer
543     open    tcp       klogin
544     open    tcp       kshell
590     open    tcp       tns-cml
659     open    tcp       unknown
661     open    tcp       unknown
816     open    tcp       unknown
980     open    tcp       unknown
984     open    tcp       unknown
988     open    tcp       unknown
993     open    tcp       simap
996     open    tcp       xtreelic
1501    open    tcp       sas-3
5520    open    tcp       sdlog
6111    open    tcp       spc
6112    open    tcp       dtspc

TCP Sequence Prediction:  Class=64K rule
                          Difficulty=1 (Trivial joke)
```

```
Remote operating system guess: HP-UX 10.20 E 9000/777 or A
712/60 with tcp random_seq = 0

Nmap run completed    1 IP address (1 host up) scanned in 2
seconds
```

Sistema Windows NT

Descripción

Este reporte fue generado de una estación de trabajo NT. Los servicios identificados no concuerdan con ninguno de los servicios UNIX comunes.

Resultados del reporte

```
Starting nmap V. 2.12 by Fyodor (fyodor@dhp.com, www.insecure.
org/nmap/)
Interesting ports on (10.0.2.124):
Port    State    Protocol   Service
135     open     tcp        loc srv
139     open     tcp        netbios ssn
427     open     tcp        svrloc

TCP Sequence Prediction: Class trivial time dependency
                         Difficulty=13 (Easy)
Remote operating system guess: Windows NT4 / Win95 / Win98

Nmap run completed    1 IP address (1 host up) scanned in 1 second
```

Sistema Cisco Router

Descripción

Este reporte muestra cuáles servicios de un enrutador Cisco están disponibles.

Resultados del reporte

```
[root@didymus /root]# nmap -O 199.82.24.1

Starting nmap V. 2.12 by Fyodor (fyodor@dhp.com,
www.insecure.org/nmap/)
Interesting ports on (10.0.2.1):
Port    State    Protocol   Service
7       open     tcp        echo
9       open     tcp        discard
13      open     tcp        daytime
19      open     tcp        chargen
23      open     tcp        telnet
79      open     tcp        finger
6001    open     tcp        X11:1
```

```
TCP Sequence Prediction: Class=random positive increments
                         Difficulty=659 (Medium)
Remote operating system guess: IOS Version 10.3(15) - 11.1(20)

Nmap run completed -- 1 IP address (1 host up) scanned in 4 seconds
```

Herramienta xtraceroute

La herramienta **xtraceroute** es una aplicación XWindow basada en su contraparte de línea de comandos, **traceroute**. Proporciona una interfaz GUI y caracteriza un globo rotatorio mostrando la ruta de una sesión **xtraceroute**. Recuérdese que el comando **traceroute**, descrito en el capítulo anterior, usa el protocolo de Internet para registrar cada destino o salto en la ruta a un *host* objetivo específico. La versión de línea de comandos registra cada salto presentando la dirección IP e información general de rendimiento. Esta versión de la herramienta proporciona la misma información pero también presenta saltos en un globo tridimensional que puede ser movido con el ratón para controlar el ángulo de visión.

El software **xtraceroute** mantiene una base de datos que contiene información del destino, como IP y el nombre del *host*. También contiene información de latitud y longitud para que, si **xtraceroute** visita ese sitio, pueda ser presentado en el globo. El usuario también tiene la capacidad de entrar destinos adicionales que no han sido proporcionados en la base de datos estándar.

La figura 5.20 muestra lo que se presentará cuando **xtraceroute** se emplee para rastrear la vía desde la Costa Este (Orlandó, FL) de los Estados Unidos de Norteamérica al sitio Web **xtraceroute** que está localizado en la Costa Oeste de Suecia. Nótese que se dibuja una línea desde Florida sobre el Atlántico y hasta Europa.

La ventana principal de la herramienta contiene dos menús, dos paneles y dos botones. El primer menú, File, tiene dos elementos: New... y Quit. El botón `Info` presenta la información específica de la entrada *host*. El otro botón, el cual está en blanco, está localizado en la ventana principal y es un atajo para el submenú New... El elemento de submenú New... se usa para entrar un destino objetivo. Para determinar la vía correcta, **xtraceroute** usará las direcciones de destino ingresadas por el usuario. Al seleccionar este submenú se presentará la ventana de diálogo que se muestra en la figura 5.21. El usuario puede entrar

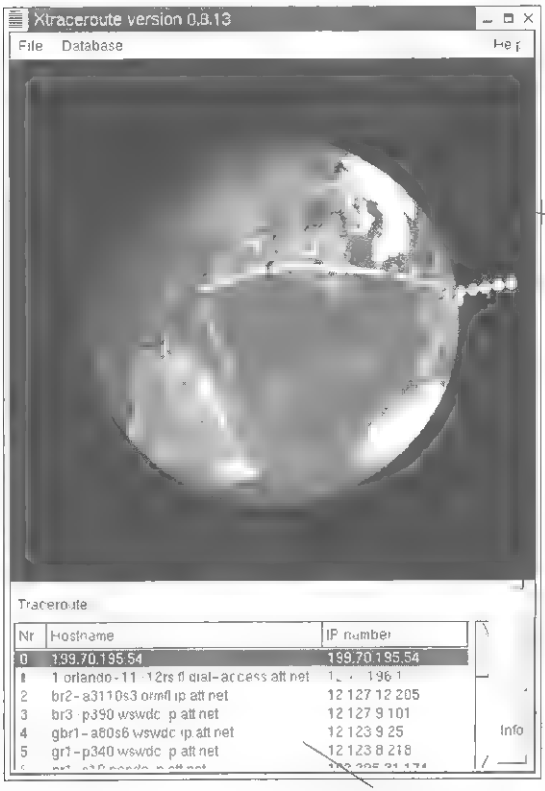


Figura 5.20 Ventana de filtro de paquete TCP.

un nombre de *host* completamente calificado o una dirección IP válida. En la ventana que se muestra en la figura 5.21, se ingresó la localización del *host* de Internet donde el software **xtraceroute** está disponible.

El otro elemento del submenú es `Quit`, el cual se explica por sí solo. El otro elemento del menú, `Database`, contiene tres elementos de submenú. Cada uno de estos elementos cumple la misma tarea básica: añadir información adicional del destino para que **xtraceroute** pueda presentar el sitio en el globo. La más útil de las tres es el elemento del submenú Add Host... (Agregar...).

Figura 5.21 Ventana del nuevo trazado.

Si selecciona esta opción, se presentará el cuadro de diálogo que se muestra en la figura 5.22.

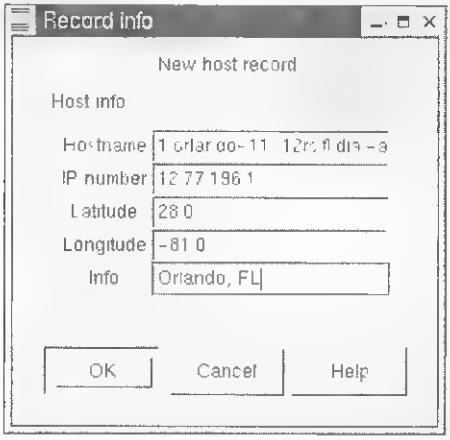


Figura 5.22 Cuadro de diálogo de registro de nuevo *host* (*new host record*).

Al dar clic en el botón `Help` en este cuadro de diálogo se obtendrá información adicional acerca de las coordenadas globales que se están agregando para la entrada de *host* que se está adicionando.

Cuando **xtraceroute** se instaló por primera vez, la base de datos no contenía ninguna información en cuanto a las redes que cubrían el área de Orlando. Como resultado, la información del *host* se adicionó manualmente a la base de datos **xtraceroute**. Parte de esta información depende de la localización en Internet del usuario. El usuario usará la ventana de registro de Nuevo *host* para agregar nuevas entradas a la base de datos. En la figura 5.22, el punto de entrada para Internet comienza con un ISP local, con

el nombre de host `1.orlando-11-12rs.fl.dial-access.att.net` y una dirección IP `12.77.196.1`, los cuales fueron ingresados en los campos de `Hostname` e `IP number`, respectivamente. Los campos de `Latitude` y `Longitude` incluyen las coordenadas globales de la localización, como se muestra en un mapa mundi o globo. Debe tenerse en cuenta que las cifras de latitud y longitud fueron aproximadas para este ejemplo. Finalmente, la localización, incluidos la ciudad y el estado, se ingresó en el campo Info.

Dar clic en el botón `OK` adicionará esta entrada a un archivo local llamado `user_hosts.cache`, el cual es almacenado en el subdirectorio `.xt` en el directorio hogar del usuario. Desafortunadamente, el directorio `.xt` debe crearse manualmente antes de que **xtraceroute** guarde cualesquiera archivos en esta localización. Este archivo se lee cuando **xtraceroute** arranca, y **xtraceroute** usa esta base de datos cuando actualiza el panel del globo tridimensional.

Como en el **traceroute** regular, el conteo de saltos y el nombre de *host*/IP de los saltos se presentan en el panel inferior de la ventana principal. A diferencia del **traceroute** normal, **xtraceroute** no proporciona información del rendimiento del viaje de ida y vuelta en la red, en el panel principal. En su lugar, el usuario debe seleccionar un salto de destino y dar clic en el botón `Info` localizado en la parte inferior del panel principal, para obtener los datos del tiempo de viaje. Dar clic en este botón presenta una nueva ventana, como se muestra en la figura 5.23.

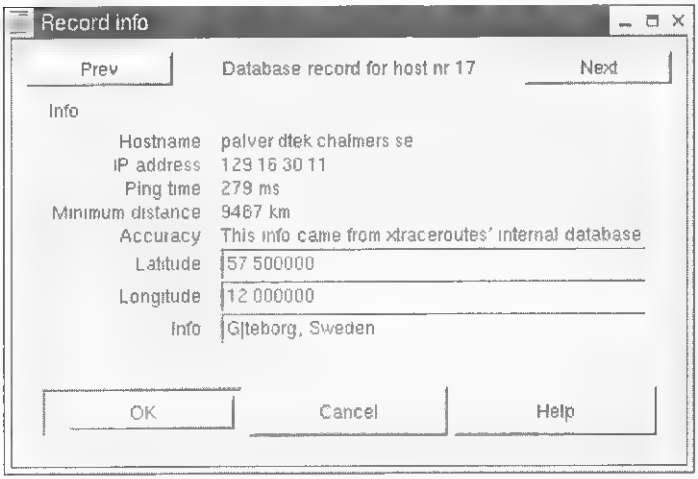


Figura 5.23 Información de salto de destino.

Esta ventana contiene unos cuantos campos que no se muestran en la figura 5.22. Estos nuevos campos incluyen los campos `Ping time`, `Minimum distance` y `Accuracy`. El campo `Ping time` representa el retardo en el viaje de ida y vuelta en microsegundos medido desde el sitio de origen, y en este caso, el destino resulta ser `palver.dtek.cnalmers.se`. Este *host* en particular está localizado en la Universidad de Tecnología de Chalmers. El campo `Minimum distance` ayuda a definir la longitud de la ruta al destino remoto. El último campo, `Accuracy`, proporciona información útil acerca de la fuente de la información del destino. La ventana también contiene dos botones que facilitan navegar por los saltos en la ventana principal. El botón `Next` acerca más al destino, mientras el botón `Prev` aleja del destino, pero acerca a la fuente del trazado.

Una de las características poderosas de **xtraceroute** es su capacidad de manipular el globo tridimensional usando el ratón. Por defecto, cuando la aplicación se arranca por primera vez, el continente completo de África está a la vista de manera predeterminada, con partes de Europa hacia la parte superior, y Suramérica en la izquierda. Cuando se ingresa un destino y **xtraceroute** comienza, dibuja un punto de inicio indicando el origen del trazado y una o más líneas indicando cada salto a lo largo de la ruta hasta el destino final. Dependiendo de cuál continente emite el usuario el trazado, el destino que conecta el punto puede no estar a la vista. En esta situación, se usa el ratón para traer las líneas a la vista una vez el trazado se ha completado. El botón izquierdo del ratón se usa para arrastrar el globo a la localización deseada. A veces, si el usuario mantiene el ratón por mucho tiempo o si el globo se arrastra muy lejos, se salta la posición deseada. Con el tiempo, y con un poco de práctica, controlar la posición de los continentes será fácil.

Como se mencionó previamente, la base de datos **xtraceroute** puede no contener el sitio donde el usuario está localizado, ningún destino a lo largo de la ruta del objetivo, o el objetivo mismo. En este caso, el usuario debe ingresar esta información manualmente. En lugar de usar la GUI, el usuario puede agregar manualmente entradas a la base de datos predeterminada localizada en su directorio hogar (*home directory*). Por ejemplo, el listado 5.2 muestra unas cuantas entradas que se usaron para proporcionar la ruta entre la Costa Este de los Estados Unidos de Norteamérica y la Costa Oeste de Suecia.

Listado 5.2 Base de datos personal del usuario para **xtraceroute.**

```
1 199.70.195.54 199.70.195.54 30 0 0n 81 0 0w
2 12.77.196.1 1.orlando-11-12rs.fl.dial-access.att.net 30
   0 0n 81 0 0w
3 12.127.9.101 br3-p390.wswdc.ip.att.net 23 33 16n 81 33
   18w
4 204.70.10.150 core5-hssi5-0-0.Washington.cw.net 38 30 0n
   77 5 59w
```

El formato de la base de datos personal incluye la dirección IP y el nombre del *host* destino, seguidos por las coordenadas globales estándar. La interpretación de las coordenadas incluidas en la primera entrada es 30 grados, cero minutos, cero segundos norte del ecuador por 81 grados, cero minutos, cero segundos en el hemisferio occidental. Cuando se agrega cada salto de destino a este archivo, **xtraceroute** presentará una transformación más precisa entre la fuente del trazado y el destino final.

06

Vista general de MIB-II

Este capítulo presenta una vista detallada de SNMP MIB-II. Como MIB-II es un estándar, los objetos están disponibles a parte de una amplia variedad de fuentes, incluidos 3Com, Brocade, Cisco y muchas otras. De hecho, cualquier dispositivo que promulgue soportar el estándar SNMP, debe soportar MIB-II. Se puede obtener una cantidad significativa de información acerca de un dispositivo particular o de un grupo de dispositivos, consultando objetos MIB-II específicos. Los objetos contenidos en las MIB suministran tanto la configuración del sistema como la información de rendimiento de la red.

Vista general de MIB-II

Los estándares SNMP definen una colección de objetos MIB (Base de Información de Administración) que soporta cada agente. El primer conjunto de objetos era conocido como MIB I, y estaban documentados en RFC1156. Con el pasar del tiempo, estos objetos fueron expandidos, y la serie de objetos recibió el nombre de MIB-II, descritos en RFC1213. Los objetos MIB-II suministran información genérica acerca de aspectos relacionados con el estado de los dispositivos del trabajo en red. Esta MIB está dividida en una colección de grupos, que se describen en la tabla 6.1.

Tabla 6.1 Lista de grupos MIB-II.

Grupo	Descripción
`system`	Suministra toda la información acerca del dispositivo o del sistema en el cual el agente está corriendo
`interfaces`	Contiene información acerca de las interfaces de operación de la red en el sistema
`at`	Tabla de traducción de dirección para direcciones Internet IP a direcciones de enlace de datos (*datalink*). Nótese que éste es un grupo depreciado
`ip`	Contiene información estadística acerca del protocolo de Internet (IP) del dispositivo
`icmp`	Contiene información estadística acerca del protocolo de mensajes de control de Internet (ICMP) del dispositivo
`tcp`	Contiene información estadística acerca del protocolo de control de transmisión (TCP) del dispositivo
`udp`	Contiene información estadística acerca del protocolo de datagramas de usuario (UDP) del dispositivo

Tabla 6.1 Lista de grupos MIB-II. (*continuación*)

Grupo	Descripción
`egp`	Contiene información estadística acerca del protocolo de puerta de enlace exterior (EGP) del dispositivo
`dot3`	Suministra información respecto a los protocolos de transmisión y acceso para cada interfaz de red
`snmp`	Contiene información estadística acerca del protocolo simple de administración de la red del dispositivo

La colección de grupos MIB II también puede presentarse gráficamente, como aparece en la figura 6.1 El grupo `mib 2` aparece como una estructura en árbol con miembros del grupo que se ramifican hacia la derecha. El número asociado o índice para cada objeto identifica la ubicación en la jerarquía del árbol. Cada uno de los grupos `mib 2` asociados, descritos en este capítulo, tiene subgrupos u objetos adicionales por debajo de él, y se presentan en el formato de vista de árbol. Recuérdese que en el capítulo 2, "Protocolo simple de administración de red", se expresó que el uso de círculos cercanos al objeto representa un subgrupo (donde pueden definirse subgrupos adicionales u objetos individuales), mientras los cuadrados representan objetos discretos individuales. Esto facilita distinguir entre una colección de objetos y objetos individuales.

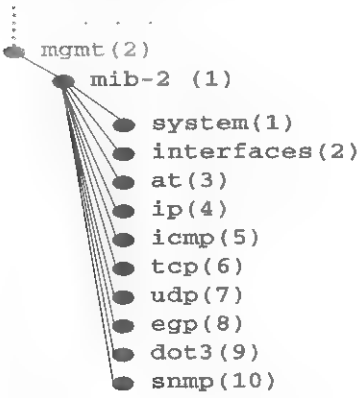


Figura 6.1 Vista del grupo `mib-2`.

Grupo system

El grupo system consta de objetos que suministran información genérica acerca del dispositivo o del sistema en el cual está co rriendo el agente. La figura 6.2 presenta una vista jerárquica del grupo system.

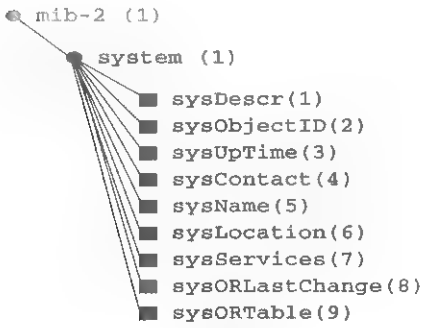


Figura 6.2 Vista del grupo system.

Los objetos de este grupo, en su mayoría, se explican por sí mismos. Los objetos sysServices, sysUpTime y SysOR prefixed requieren información adicional. El objeto SysServices representa un código de siete bits, que corresponde al valor de los servicios combinados que suministra el dispositivo. Cada bit del código está relacionado con una de las capas del modelo OSI, y si el dispositivo ofrece un servicio en una capa particular, entonces se establece el bit para dicha capa. Por ejemplo, considérese un dispositivo que suministra funciones de enrutado. El valor asociado sysServices para este dispositivo es 72. La tabla 6.2 contiene una lista de capas y servicios asociados. El valor de sysServices es la suma de los valores de los bits, donde el valor de cualquier bits asociado es 2 elevado a la potencia L-1 (donde L es la capa). Por tanto, para un dispositivo conmutador que es un dispositivo de capas 1 y 2, tenemos:

$$2^{1-1} + 2^{2-1} = 3$$

y/o para un servidor de aplicación (capas 4 y 7), tenemos:

$$2^{4-1} + 2^{7-1} = 72$$

Incluimos la capa 4 en esta ecuación porque un servidor de aplicación suministra servicios en ambas capas, 7 y 4.

Nota Actualmente no existen definiciones para las capas 5 y 6.

Tabla 6.2 Capas funcionales utilizadas para determinar `sysServices`.

Capa	Funcionalidad de dispositivo
1	Física: un dispositivo que opera en esta capa se conoce como un repetidor de red
2	Capa de enlace de datos (*datalink*) y/o de subred: un dispositivo que opera en esta capa incluye un puente de red o conmutador
3	Internet/red: un dispositivo que opera en esta capa es una puerta de enlace o un enrutador
4	Servicios de extremo a extremo, como *host* IP
7	Servicios de aplicación, como repetidores de correo, servidor DNS, etc.

El objeto `sysUpTime` indica la cantidad de tiempo que ha transcurrido desde que el agente administrativo de la red fue arrancado por última vez. Esto no significa necesariamente que el dispositivo por sí mismo haya estado operando desde dicho momento. Tómese, por ejemplo, un agente corriendo sobre un sistema operativo. El agente puede ser rearrancado independientemente del sistema y, por tanto, puede no representar la cantidad verdadera de tiempo que el sistema ha estado operando. Sin embargo, con algunos dispositivos, el agente no puede ser arrancado o detenido de manera independiente del sistema. En este caso, el agente debe ser más preciso.

El grupo `system` también contiene unos cuantos objetos relacionados con dispositivos SNMPv2 que desempeñan el papel de agentes. Estos objetos tienen nombres que comienzan con el prefijo `sysOR` y son soportados con agentes SNMPv2 (o más modernos).

Los objetos controlan la configuración dinámica de recursos de objeto. Como estos objetos fueron introducidos con los estándares SNMPv2, los agentes SNMPv1 no los soportan.

El grupo `system` contiene el siguiente conjunto de objetos:

Nombre del objeto:	`sysDescr`
OID:	`system.1`
Tipo de objeto:	`Display String [255]`
Modo de acceso:	`sólo lectura`
Descripción:	Describe el dispositivo o entidad, como tipo de dispositivo, características de hardware, información del sistema operativo, etc.

Nombre del objeto:	`sysObjectID`
OID:	`system.2`
Tipo de objeto:	`Object Identifier`
Modo de acceso:	`sólo lectura`
Descripción:	Muestra la identificación autorizada del vendedor del dispositivo

Nombre del objeto:	`sysUpTime`
OID:	`system.3`
Tipo de objeto:	`TimeTick`
Modo de acceso:	`sólo lectura`
Descripción:	Presenta la cantidad de tiempo transcurrida desde que la porción de administración de la red del sistema (agente) fue rearrancada por última vez

Nombre del objeto:	`sysContact`
OID:	`system.4`
Tipo de objeto:	`Display String [255]`

Modo de acceso: `lectura-escritura`

Descripción: Presenta información que indica la persona con quien se hace contacto y/u otra organización que suministra soporte para este dispositivo

Nombre del objeto: `sysName`

OID: `system.5`

Tipo de objeto: `Display String [255]`

Modo de acceso: `lectura-escritura`

Descripción: Presenta el nombre del dispositivo; puede ser el nombre oficial del *host* o cualquier otro nombre asignado administrativamente

Nombre del objeto: `sysLocation`

OID: `system.6`

Tipo de objeto: `Display String [255]`

Modo de acceso: `lectura-escritura`

Descripción: Indica la ubicación física donde está instalado el dispositivo

Nombre del objeto: `sysServices`

OID: `system.7`

Tipo de objeto: `Integer`

Modo de acceso: `sólo lectura`

Descripción: Presenta los servicios que suministra este dispositivo

Nombre del objeto: `sysOrLastChange`

OID: `system.8`

Tipo de objeto:	TimeStamp
Modo de acceso:	sólo lectura
Descripción:	Muestra el valor del objeto sysUpTime en el momento del cambio más reciente hecho en cualesquier instancia del objeto sysORID

Nombre del objeto:	sysORTable
OID:	system.9
Tipo de objeto:	Sequence of sysOREntry
Modo de acceso:	sólo lectura
Descripción:	Presenta una tabla de recursos de objetos configurables dinámicamente en un sistema SNMPv2 actuando en el papel de agente

Nombre del objeto:	sysOREntry
OID:	sysORTable.1
Tipo de objeto:	Sequence
Modo de acceso:	sólo lectura
Descripción:	Presenta información sobre un objeto configurable específico

Nombre del objeto:	sysORIndex
OID:	sysOREntry.1
Tipo de objeto:	Object Indentifier
Modo de acceso:	sólo lectura
Descripción:	Se usa como índice en la sysORTable

Nombre del objeto:	sysORID
OID:	sysOREntry.2

Tipo de objeto: Display String [255]

Modo de acceso: sólo lectura

Descripción: Presenta el OID de esta entrada, análogo al objeto sysObjectID

Nombre del objeto: sysORDescr

OID: sysOREntry.3

Tipo de objeto: Display String [255]

Modo de acceso: sólo lectura

Descripción: Muestra una descripción del recurso del objeto, análogo al objeto sysDescr

Nombre del objeto: sysORUpTime

OID: sysOREntry.4

Tipo de objeto: TimeStamp

Modo de acceso: sólo lectura

Descripción: Contiene el valor del objeto sysUpTime en el momento en que esta instancia (fila) fue actualizada por última vez o creada

Cuando se consulta el grupo system de un sistema Linux, retornan los siguientes objetos con sus valores asociados:

```
system.sysDescr.0 = "Linux socrates 2.2.5-15 #5 Mon Aug 2
   21:07:04 EDT 1999 i586"
system.sysObjectID.0 = OID: enterprises.ucdavis.ucdSn-mpAgent.
   linux
system.sysUpTime.0 = Timeticks: (160910) 0:26:49.10
system.sysContact.0 = "Steve Rossman"
system.sysName.0 = "socrates"
system.sysLocation.0    "Media Lab - 10th Floor"
system.sysServices.0    72
```

En este ejemplo, el objeto `sysDescr` incluye información acerca del sistema en el cual está corriendo el agente. La cadena suministra la misma información básica que se obtiene del comando UNIX **uname**:

```
# uname -a
Linux socrates 2.2.5-15 #5 Mon Aug 2 21:07:04 EDT 1999 i586 unknown
```

El objeto `sysObject` contiene un OID de la rama Linux que identifica al agente. El `sysContact` muestra que el agente ha estado corriendo durante aproximadamente 26 minutos y 49 segundos. Los objetos `sysContact`, `sysName` y `sysLocation` contienen información específica acerca del propietario (Steve Rossman), el nombre (socrates) y la ubicación (Media Lab - 10th Floor) del dispositivo. El capítulo 7, "Uso de los agentes SNMP", nos mostrará cómo actualizar estos campos para incluir información específica en relación con su sistema y entorno. Finalmente, el objeto `sysServices` muestra que el dispositivo suministra servicios de la capa 4 y de la capa 7.

Grupo interfaces

El grupo `interfaces` suministra información tanto de configuración como estadística respecto a las interfaces de red instaladas en el dispositivo. Como se analizó en el capítulo 2, "Protocolo simple de administración de red", este grupo tiene un objeto `ifNumber`, que contiene el número total de interfaces de red instaladas en el sistema, independientemente del estado operativo de cualquier interfaz en particular. El otro objeto, `ifTable`, es una tabla que contiene un registro para cada interfaz. La tabla es indexada por el objeto `ifIndex` y contiene un valor entre 1 y el valor del objeto `ifNumber`. El número `ifIndex` puede seleccionar una columna o interfaz directamente. `ifTable` contiene 22 objetos que suministran lo siguiente:

- tipo, capacidad y otras características de interfaz
- información operacional
- información de rendimiento y estadística

La figura 6.3 muestra una representación jerárquica del grupo `interfaces`. La figura muestra diferentes grupos de objetos, que incluyen características de interfaces y contadores de tráfico entrante/saliente.

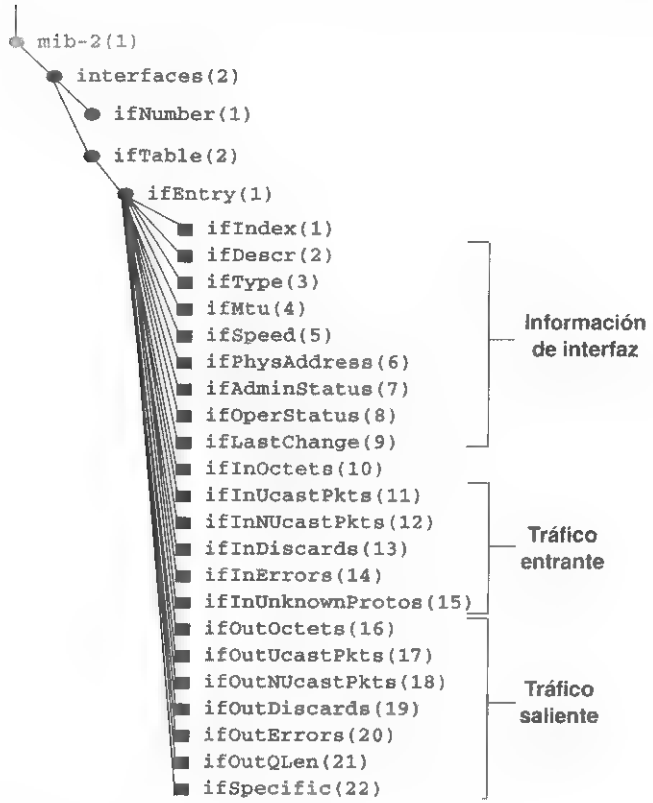


Figura 6.3 Vista del grupo `interfaces`.

Al consultar la tabla `interfaces` en un sistema Linux, se muestra una salida similar a lo siguiente:

```
interfaces.ifNumber.0 = 2
interfaces.ifTable.ifEntry.ifIndex.1 = 1
interfaces.ifTable.ifEntry.ifIndex.2 = 2
interfaces.ifTable.ifEntry.ifDescr.1 = "lo0" Hex: 6C 6F 30
interfaces.ifTable.ifEntry.ifDescr.2 = "eth0" Hex: 65 74 68 30
interfaces.ifTable.ifEntry.ifType.1 = softwareLoopback(24)
interfaces.ifTable.ifEntry.ifType.2 = ethernet-csmacd(6)
interfaces.ifTable.ifEntry.ifMtu.1   3924
interfaces.ifTable.ifEntry.ifMtu.2 = 1500
interfaces.ifTable.ifEntry.ifSpeed.1 = Gauge: 10000000
interfaces.ifTable.ifEntry.ifSpeed.2 = Gauge: 10000000
interfaces.ifTable.ifEntry.ifPhysAddress.1 = ""
interfaces.ifTable.ifEntry.ifPhysAddress.2 = Hex: 00 10 5A 28 5D 7C
```

(continúa)

```
interfaces.ifTable.ifEntry.ifAdminStatus.1 = up(1)
interfaces.ifTable.ifEntry.ifAdminStatus.2 = up(1)
interfaces.ifTable.ifEntry.ifOperStatus.1 = up(1)
interfaces.ifTable.ifEntry.ifOperStatus.2 = up(1)
interfaces.ifTable.ifEntry.ifLastChange.1 = Timeticks: (0) 0:00:00.00
interfaces.ifTable.ifEntry.ifLastChange.2 = Timeticks: (0) 0:00:00.00
interfaces.ifTable.ifEntry.ifInOctets.1 = 373912
interfaces.ifTable.ifEntry.ifInOctets.2 = 50204
interfaces.ifTable.ifEntry.ifInUcastPkts.1 = 1218
interfaces.ifTable.ifEntry.ifInUcastPkts.2 = 163
interfaces.ifTable.ifEntry.ifInNUcastPkts.1 = 0
interfaces.ifTable.ifEntry.ifInNUcastPkts.2 = 0
interfaces.ifTable.ifEntry.ifInDiscards.1 = 0
interfaces.ifTable.ifEntry.ifInDiscards.2 = 0
interfaces.ifTable.ifEntry.ifInErrors.1 = 0
interfaces.ifTable.ifEntry.ifInErrors.2 = 0
interfaces.ifTable.ifEntry.ifInUnknownProtos.1 = 0
interfaces.ifTable.ifEntry.ifInUnknownProtos.2 = 0
interfaces.ifTable.ifEntry.ifOutOctets.1 = 381304
interfaces.ifTable.ifEntry.ifOutOctets.2 = 174020
interfaces.ifTable.ifEntry.ifOutUcastPkts.1 = 1242
interfaces.ifTable.ifEntry.ifOutUcastPkts.2 = 565
interfaces.ifTable.ifEntry.ifOutNUcastPkts.1 = 0
interfaces.ifTable.ifEntry.ifOutNUcastPkts.2 = 0
interfaces.ifTable.ifEntry.ifOutDiscards.1 = 0
interfaces.ifTable.ifEntry.ifOutDiscards.2 = 0
interfaces.ifTable.ifEntry.ifOutErrors.1 = 0
interfaces.ifTable.ifEntry.ifOutErrors.2 = 0
interfaces.ifTable.ifEntry.ifOutQLen.1 = Gauge: 0
interfaces.ifTable.ifEntry.ifOutQLen.2 = Gauge: 0
interfaces.ifTable.ifEntry.ifSpecific.1 = OID: .ccitt.nullOID
interfaces.ifTable.ifEntry.ifSpecific.2 = OID: .ccitt.nullOID
```

Nótese que el objeto `ifNumber` es igual a `2` porque un total de dos interfaces están definidas en el sistema. Como resultado, los objetos `ifIndex.1` e `ifIndex.2` están colocados en `1` y `2`, respectivamente, de manera que se puedan usar para indexar cada interfaz de manera separada. El objeto `ifDescr` contiene el nombre de la interfaz como es conocido por el agente que está corriendo en el dispositivo. Por ejemplo, usando la salida anterior, la primera interfaz Ethernet se conoce como `eth0`. Como resultado, el objeto `ifDescr` contendrá este nombre de interfaz, con cada carácter convertido a hexadecimal (`"eth0" Hex: 65 74 68 30`). Así, el valor hexadecimal de `eth0` es `65 74 68 30`. Los nombres de interfaces suministrados por el agente son los mismos nombres que se presentan cuando se usan los comandos **ifconfig** o **nestat**. *Véase* el capítulo 4, "Herramientas y utilidades centrales del sistema", para obtener información adicional.

El objeto `ifType` registra el tipo de la interfaz de red usando un identificador único entero. El número se puede transferir a una cadena de palabra clave que da una información más descriptiva respecto a la interfaz que se está empleando en el momento. Una gran cantidad de tipos de interfaces de red están definidas por el estándar MIB-II. La tabla 6.3 lista algunas de las interfaces más populares. Consulte el RFC1213 para obtener una lista completa. Obsérvese que `eth0` porta el tipo `6` de interfaz, que es `ethernetCsmacd`.

El objeto `ifMtu`, o máxima unidad de transferencia (MTU), identifica el máximo tamaño de la unidad de datos del protocolo PDU (o marco), que es permitido para la interfaz. La Ethernet estándar es de `1500` MTU, mientras que en el `software_loopback` es mucho mayor de `3924`. Sistemas diferentes pueden implementar la MTU para el software *loopback*, con varios valores considerados apropiados para cada sistema. El objeto `ifSpeed` muestra la capacidad máxima de la interfaz. En el ejemplo, ambas interfaces contienen la misma velocidad de `10000000`. El valor representa el rendimiento teórico de una Ethernet LAN, que es 10Mb por segundo. Otras interfaces pueden mostrar capacidades menores o mayores, dependiendo del tipo de interfaz. Por ejemplo: una interfaz serial que soporta PPP contiene un `ifSpeed` de `9600`, que representa `9600` bits por segundo que pueden ser soportados, dadas las características de hardware de una interfaz serial RS-232.

Tabla 6.3 Tipos y descripción de interfaces comunes.

Número	Tipo	Descripción
1	other	Ninguno de los tipos siguientes
6	ethernetCsmacd	Protocolo de red Ethernet
7	iso88023Csmacd	IEEE 802.3 Carrier Sense Protocolo de red de acceso múltiple con detección de colisión (CSMA/CD)
9	iso88025TokenRing	IEE 802.5 Protocolo de red TokenRing
15	fddi	Protocolo de red de interfaz (FDDI) de datos distribuidos de fibra ANSI
20	basicISDN	Red digital de servicios integrados (ISDN) de tasa básica
21	primaryISDN	Tasa primaria ISDN
23	ppp	Protocolo de Internet punto a punto
23	softareLoopback	Interfaz interna empleada para comunicaciones entre procesos
30	ds3	Línea de transmisión digital que utiliza el formato DS-3
37	atm	Modo de transferencia asincrónica (ATM)
46	hssi	Interfaz serial de alta velocidad (HSSI)

El objeto `ifPhysAddress` identifica la dirección del protocolo de enlace de datos (cuando es apropiado) para la interfaz. La interfaz `eth0` tiene un `ifPhysAddress` de `00 10 5A 28 5D 7C`, mientras que el `ifPhysAddress` contiene un valor de cadena

nulo (""). La razón para esto es que el software de *loopback* no utiliza ningún hardware, y no se necesita ninguna dirección de enlace de datos. La dirección que se encuentra en el `ifPhysAddress` se usa para comunicaciones en red de bajo nivel entre sistemas. Cada vez que se emite un paquete desde esta interfaz, esta dirección se usa como fuente del paquete.

Los objetos `ifAdminStatus` e `ifOperStatus` muestran el estado administrativo y el estado operacional de la interfaz. El administrador de la red utiliza el estado administrativo para controlar la interfaz. Este objeto proporciona la capacidad para controlar cuando la interfaz es marcada como arriba (`up`) o abajo (`down`). Además, se puede establecer `testing`, un tercer estado. El estado `up` significa que se les permite a los paquetes fluir a través de la interfaz, mientras que el estado `down` implica que ningún paquete se va a recibir o a enviar desde esta interfaz. Esto es independiente del estado de la conexión física a la interfaz. En otras palabras, una interfaz de red puede estar conectada a una red operativa, pero si es marcada `down`, ningún tráfico de la red será leído por la interfaz. El estado `testing` habilita el diagnóstico interno de la interfaz para validar la operación correcta de la interfaz. El objeto `ifOperStatus` muestra el estado actual de la interfaz, que es uno de los estados definidos representado por el objeto `ifAdminStatus`. Este objeto obtiene el estado de una interfaz particular.

Nota

El acceso del objeto `ifAdminStatus` es lectura-escritura (*read-write*); todos los demás objetos en el grupo interfaces no pueden ser modificados por un administrador de red, porque tienen acceso de sólo lectura (*read-only*). La razón para esto podría parecer bastante natural y sencilla. Es razonable que los contadores y la información descriptiva acerca de una interfaz no deban cambiarse, porque es importante mantener la información del tipo de interfaz, con el fin de evitar problemas de configuración del trabajo en red y mantener una medida de rendimiento exacta.

Los objetos `ifSpecific` contienen el valor de `ccitt.nullOID`, que representa una cadena OID válida pero nula. El grupo `interfaces` contiene el siguiente conjunto de objetos:

Nombre del objeto:	`IfNumber`
OID:	`interfaces.1`
Tipo de objeto:	`Integer`
Modo de acceso:	`sólo lectura`
Descripción:	Número total de interfaces de la red contenidas en el sistema local

Nombre del objeto:	`ifTable`
OID:	`interfaces.2`
Tipo de objeto:	`Sequence of ifEntry`
Modo de acceso:	`na`
Descripción:	Lista o fila de entradas de interfaz para esta tabla

Nombre del objeto:	`ifEntry`
OID:	`interface.ifTable`
Tipo de objeto:	`Sequence`
Modo de acceso:	`not accessible`
Descripción:	Entrada de interfaz específica que contiene todos los objetos definidos enseguida

Nombre del objeto:	`ifIndex`
OID:	`ifEntry.1`
Tipo de objeto:	`Integer`
Modo de acceso:	`sólo lectura`
Descripción:	Definición de referencia MIB que es específica para un tipo de medio particular que se usa para tener acceso a la interfaz de red

Nombre del objeto:	`IfDescr`
OID:	`ifEntry.2`
Tipo de objeto:	`DisplayString [255]`
Modo de acceso:	`sólo lectura`
Descripción:	Contiene una cadena de descripción de la interfaz que incluye el nombre de la interfaz desde el punto de vista de un sistema operativo; los posibles valores incluyen: `eth0`, `ppp0` y `lo0`

Nombre del objeto:	`ifType`
OID:	`ifEntry.3`
Tipo de objeto:	`DisplayString (255)`
Modo de acceso:	`sólo lectura`
Descripción:	El tipo de interfaz. La tabla 6.3 lista tipos específicos

Nombre del objeto:	`ifMtu`
OID:	`ifEntry.4`
Tipo de objeto:	`Integer`
Modo de acceso:	`sólo lectura`
Descripción:	Unidad máxima de transmisión de la interfaz. Esto representa el marco más grande que se puede enviar y/o recibir en la interfaz

Nombre del objeto:	`ifSpeed`
OID:	`ifEntry.5`
Tipo de objeto:	`Gauge`
Modo de acceso:	`sólo lectura`
Descripción:	Tasa de datos (capacidad) de la interfaz

Nombre del objeto:	`IfPhysAddress`
OID:	`ifEntry.6`
Tipo de objeto:	`PhysAddress`
Modo de acceso:	`sólo lectura`
Descripción:	Dirección de enlace de datos de la interfaz

Nombre del objeto:	`ifAdminStatus`
OID:	`ifEntry.7`
Tipo de objeto:	`Integer`
Modo de acceso:	`sólo lectura`
Descripción:	Estado administrativo de la interfaz, que es uno de los estados definidos y listados en el objeto `ifOperStatus`. El propietario del dispositivo puede controlar la interfaz con este objeto

Nombre del objeto:	`ifOperStatus`
OID:	`ifEntry.8`
Tipo de objeto:	`Integer`
Modo de acceso:	`sólo lectura`
Descripción:	Estado operacional de la interfaz. Los estados definidos incluyen `up(1)`, `down(2)` y `testing(3)`

Nombre del objeto:	`ifLastChange`
OID:	`ifEntry.9`
Tipo de objeto:	`TimeTicks`
Modo de acceso:	`sólo lectura`

Descripción:	Momento en que la interfaz fue actualizada por última vez a su actual estado operativo
Nombre del objeto:	`ifInOctets`
OID:	`ifEntry.10`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de octetos (bytes) recibidos en la interfaz, incluidos cualesquiera bytes de enmarcado de enlace de datos
Nombre del objeto:	`ifInUcastPkts`
OID:	`ifEntry.11`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de paquetes de unidifusión (*unicast*) enviados mediante un protocolo de mayor nivel a una subred
Nombre del objeto:	`ifINUcastPkts`
OID:	`ifEntry.12`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de paquetes que no son de unidifusión, entregados a un protocolo de mayor nivel de trabajo en red
Nombre del objeto:	`ifInDiscards`
OID:	`ifEntry.13`

Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de paquetes entrantes descartados (a pesar de que no tenían errores) y que no serán enviados a un protocolo de mayor nivel de trabajo en red

Nombre del objeto:	`ifInErrors`
OID:	`ifEntry.14`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de paquetes entrantes con errores que ocasionaron que no fueran entregados a un protocolo de mayor nivel de trabajo en red

Nombre del objeto:	`ifInUnknownProtos`
OID:	`ifEntry.15`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de paquetes entrantes descartados, debido a un protocolo de trabajo en red desconocido o no soportado

Nombre del objeto:	`ifOutOctets`
OID:	`ifEntry.16`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de octetos (bytes) transmitidos en la interfaz. Esto incluye también cualesquiera bytes de enmarcado de enlace de datos

Nombre del objeto:	`IfOutUcastPkts`
OID:	`ifEntry.17`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de paquetes que protocolos de alto nivel (como IP) solicitaron que fueran transmitidos a una dirección de unidifusión de red. Esto incluye aquellos que han sido descartados o no enviados por cualquier motivo

Nombre del objeto:	`ifOutNUcastPkts`
OID:	`ifEntry.18`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de paquetes que protocolos de alto nivel (como IP) solicitaron que fueran transmitidos a una dirección de no unidifusión. Esto también incluye aquellos paquetes que han sido descartados, o por alguna otra razón, no enviados

Nombre del objeto:	`ifOutDiscards`
OID:	`ifEntry.19`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de paquetes que no pudieron ser transmitidos, debido a alguna razón no relacionada con una condición de error específica. Esto pudo haber sido causado, por ejemplo, por el TTL de un paquete

Nombre del objeto:	`ifOutErrors`
OID:	`ifEntry.20`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de paquetes que no pudieron ser transmitidos debido a errores

Nombre del objeto:	`ifOutQ_en`
OID:	`ifEntry.21`
Tipo de objeto:	`Gauge`
Modo de acceso:	`sólo lectura`
Descripción:	Longitud de la cola de salida del paquete en el dispositivo

Nombre del objeto:	`ifSpecific`
OID:	`ifEntry.22`
Tipo de objeto:	`Object Identifier`
Modo de acceso:	`sólo lectura`
Descripción:	Definición de referencia MIB que es específica para el tipo particular de medio empleado para llevar a cabo la interfaz de trabajo en red

Grupo de traducción de direcciones

El grupo `at` consta de una tabla sencilla que suministra transformación de una dirección de la red a una dirección física. La figura 6.4 muestra una vista de árbol del grupo de traducción de direcciones. Este grupo ha sido depreciado (lo que significa que ha sido remplazado por otro grupo de objetos MIB), pero se suministra para compatibilidad hacia atrás con el estándar existente MIB-I. Muchos agentes existentes continúan soportando este grupo, incluido el agente Linux. Ya que éste es el caso, el grupo se describe a continuación.

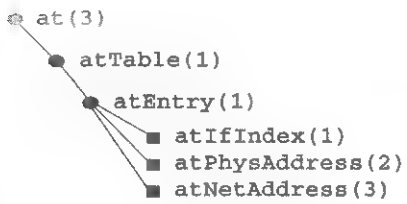


Figura 6.4 Vista del grupo `at`.

Cada fila de la tabla `at` suministra una transformación entre una dirección de red y una dirección física. Al consultar la tabla `at` en un sistema Linux, presenta una salida similar a la siguiente:

```
at.atTable.atEntry.atIfIndex.1.1.10.0.2.10 = 1
at.atTable.atEntry.atIfIndex.1.1.10.0.2.50 = 1
at.atTable.atEntry.atIfIndex.1.1.10.0.2.127 = 1
at.atTable.atEntry.atPhysAddress.1.1.10.0.2.10   Hex: 00 00 00 00 00 00
at.atTable.atEntry.atPhysAddress.1.1.10.0.2.50 = Hex: 00 C0 D4 00 09 40
at.atTable.atEntry.atPhysAddress.1.1.10.0.2.127 = Hex: 00 60 97 0E A3 06
at.atTable.atEntry.atNetAddress.1.1.10.0.2.10 = IpAddress: 10.0.2.10
at.atTable.atEntry.atNetAddress.1.1.10.0.2.50 = IpAddress: 10.0.2.50
at.atTable.atEntry.atNetAddress.1.1.10.0.2.127 = IpAddress: 10.0.2.127
```

La dirección de red es típicamente la dirección IP asignada a la interfaz. La dirección física representa la dirección de red de enlace de datos apropiada para dicha interfaz. La tabla es indexada por el objeto `atIfIndex`, el cual contiene una transformación para el objeto `ifIndex` contenido en el grupo `Interfaces`. El objeto de dirección de red también indexa la tabla `at`. Por ejemplo, `atIfIndex.1.1.10.0.2.50` se usa como un puntero hacia los objetos `atPhysAddress` y `atNetAddress` asociados con esa dirección IP. Nótese que la cadena OID contiene la dirección IP actual de la interfaz y sirve como llave de índice. El objeto `atPhysAddress` contiene la dirección física para el protocolo de enlace de datos para ese tipo de interfaz. El ejemplo muestra una dirección Ethernet válida en hexadecimal. El objeto `atNetAddress` contiene la dirección IP válida para la interfaz, expresada en notación normal decimal punteada.

La tabla `at` cambiará de tamaño de manera concertada con el caché ARP. Dependiendo del tráfico en la red, la tabla ARP crecerá y se reducirá. El comando `arp` puede listar la tabla ARP en un sistema UNIX. El capítulo 4, "Herramientas y utilidades centrales del sistema", explica completamente esta utilidad. Además, como la tabla `at` representa las entradas ARP, esta lista también

transformaciones incompletas. Esto se puede observar examinando el listado `atPhysAddress.1.1.10.0.2.10 = Hex: 00 00 00 00 00 00`. Nótese que el valor hexadecimal es todo ceros, lo que implica que no se ha obtenido ninguna transformación de esta dirección IP.

La tabla `at` contiene los siguientes objetos:

Nombre del objeto:	`atTable` **(depreciado)**
OID:	`mib-2.at`
Tipo de objeto:	`Sequence of AtEntry`
Modo de acceso:	`sólo lectura`
Descripción:	Transformación entre direcciones de red y direcciones físicas (de enlace de datos)

Nombre del objeto:	`ifEntry` **(depreciado)**
OID:	`mib-2.at`
Tipo de objeto:	`Sequence`
Modo de acceso:	`sólo lectura`
Descripción:	Transformación específica que contiene todos los objetos restantes listados enseguida

Nombre del objeto:	`atIfIndex` **(depreciado)**
OID:	`mib-2.at`
Tipo de objeto:	`Integer`
Modo de acceso:	`lectura-escritura`
Descripción:	Referencia a cada transformación específica

Nombre del objeto:	`atPhysAddress` **(depreciado)**
OID:	`ifEntry.22`
Tipo de objeto:	`PhysAddress`

Modo de acceso:	`lectura-escritura`
Descripción:	Una dirección física dependiente del medio (que es una dirección IP válida)

Nombre del objeto:	`atNetAddress` **(depreciado)**
OID:	`ifEntry.22`
Tipo de objeto:	`NetworkAddress`
Modo de acceso:	`lectura-escritura`
Descripción:	La dirección IP de la dirección física dependiente del medio asociado

Grupo de protocolo de Internet

El grupo `ip` contiene contadores de la red que miden el flujo de tráfico IP hacia el interior y el exterior del sistema. El grupo también contiene tablas adicionales que suministran enrutado a la capa de red e información de transformación de enlace de datos. La figura 6.5 muestra el grupo `ip` en la vista tipo árbol.

El grupo `ip` contiene el siguiente conjunto de objetos:

Nombre del objeto:	`ipForwarding`
OID:	`ip.1`
Tipo de objeto:	`Integer`
Modo de acceso:	`lectura-escritura`
Descripción:	Indica si el sistema está actuando como una puerta de enlace (enrutador IP) o solo como un *host* corriente que no suministra servicios de reenvío. Los valores válidos son `Forwarding(1)` y `notForwarding(2)`

Nombre del objeto:	`ipDefaultTTL`
OID:	`ip.2`
Tipo de objeto:	`Integer`

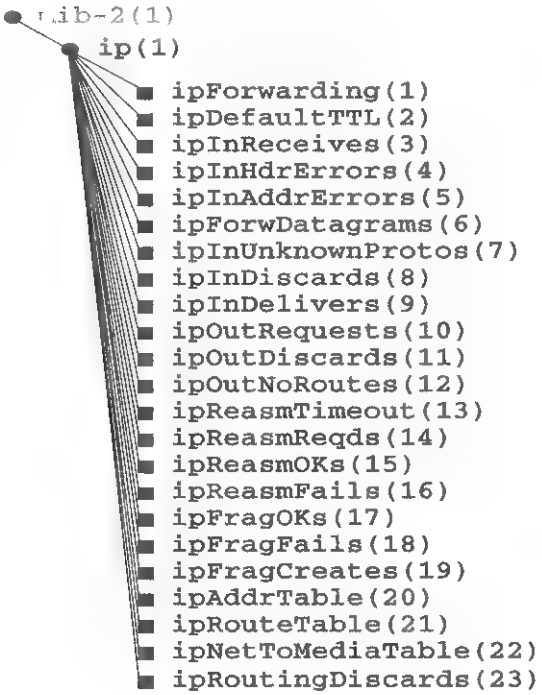


Figura 6.5 Vista del grupo `ip`.

Modo de acceso:	`lectura-escritura`
Descripción:	Valor de tiempo de vida (TTL) que es colocado en el campo TTL de un paquete IP

Nombre del objeto:	`ipInReceives`
OID:	`ip.3`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número total de paquetes de entrada recibidos desde todas las interfaces que operan en el sistema

Nombre del objeto:	`ipInHdrError`

OID:	`ip.4`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de paquetes de entrada que fueron rechazados como resultado de un error en el encabezamiento IP

Nombre del objeto:	`ipInaddrErrors`
OID:	`ip.5`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de paquetes de entrada que fueron rechazados porque la dirección de destino final IP era inválida para este sistema

Nombre del objeto:	`ipForwDatagrams`
OID:	`ip.6`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de paquetes que el sistema local intentó reenviar cuando actuaba como puerta de enlace o enrutador

Nombre del objeto:	`ipInUnknownProtos`
OID:	`ip.7`
Tipo de objeto:	`NetworkAddress`
Modo de acceso:	`lectura-escritura`
Descripción:	Número de paquetes que fueron recibidos de la red, exitosamente descartados porque el protocolo de capa de la red no era soportado o porque el sistema no lo conocía

Nombre del objeto:	`ipInDiscards`
OID:	`ip.8`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de paquetes de entrada que fueron rechazados, debido a una falta de espacio en el buffer o por otra condición no relacionada con el paquete en sí mismo

Nombre del objeto:	`ipInDelivers`
OID:	`ip.9`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de paquetes de entrada que fueron enviados exitosamente a protocolos de mayor nivel

Nombre del objeto:	`ipOutRequests`
OID:	`ip.10`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de paquetes IP que protocolos de mayor nivel enviaron al protocolo IP para transmisión

Nombre del objeto:	`ipOutDiscards`
OID:	`ip.11`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`

Descripción: Número de paquetes que fueron descartados, debido a falta de espacio en el buffer o a otra condición que no está relacionada con el paquete mismo

Nombre del objeto: `ipOutNoRoutes`
OID: `ip.12`
Tipo de objeto: `Counter`
Modo de acceso: `sólo lectura`
Descripción: Número de paquetes que fueron descartados debido a que no se conocía ninguna ruta hacia la red de destino solicitada

Nombre del objeto: `ipReasmTimeout`
OID: `ip.13`
Tipo de objeto: `Counter`
Modo de acceso: `sólo lectura`
Descripción: Intervalo en segundos que los paquetes del fragmento de entrada IP son retenidos antes de ser rearmados o reensamblados

Nombre del objeto: `ipReasmReqds`
OID: `ip.14`
Tipo de objeto: `Counter`
Modo de acceso: `sólo lectura`
Descripción: Número de fragmentos de paquetes IP recibidos que deben ser reensamblados

Nombre del objeto:	`ipReasmOKs`
OID:	`ip.15`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de fragmentos de paquetes IP que fueron reensamblados exitosamente

Nombre del objeto:	`ipReasmFails`
OID:	`ip.16`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de fallas en reensamblaje detectadas

Nombre del objeto:	`ipFragOK`
OID:	`ip.17`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de paquetes que fueron fragmentados exitosamente

Nombre del objeto:	`ipFragsFails`
OID:	`ip.18`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de paquetes que no fueron fragmentados porque la bandera `don't-fragment` que se encuentra en el encabezamiento del IP estaba activada

Nombre del objeto:	`ipFragsCreates`
OID:	`ip.19`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de fragmentos de paquetes IP generados en el sistema

Nombre del objeto:	`ipAddrTable`
OID:	`ip.20`
Tipo de objeto:	`Sequence of IpAddrEntry`
Modo de acceso:	`not-accessible`
Descripción:	Tabla de información de direccionamiento que está relacionada con las direcciones IP del sistema

Nombre del objeto:	`ipRouteTable`
OID:	`ip.21`
Tipo de objeto:	`Sequence Of IpRouteEntry`
Modo de acceso:	`not-accessible`
Descripción:	Una ruta a un destino particular

Nombre del objeto:	`ipNetToMediaTable`
OID:	`ip.22`
Tipo de objeto:	`Sequence Of ipNetToMediaEntry`
Modo de acceso:	`not-accessible`
Descripción:	Transformación entre direcciones IP y direcciones de vínculos de datos

Nombre del objeto:	`ipRoutingDiscards`
OID:	`ip.23`

Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de paquetes que fueron descartados, a pesar de que eran válidos

En el grupo `ip` se incluyen tres tablas: `ipAddrTable`, `ipRouteTable` e `ipNetToMediaTable`. La tabla `ipAddrTable` almacena información relacionada con las direcciones IP utilizadas en el sistema. La figura 6.6 presenta la vista en árbol de esta tabla.

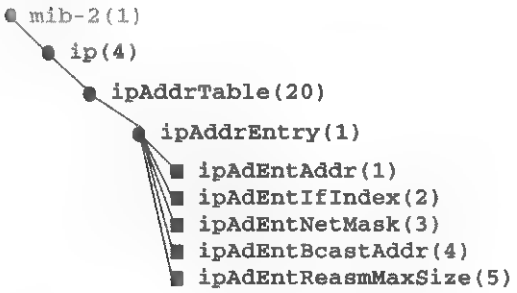


Figura 6.6 Vista de la tabla `ipAddrTable`.

A cada fila de la tabla, que es indexada por el objeto `ipAdEntIfIndex` e incluye una única dirección IP que corresponde a la interfaz física de la red, se le asigna el número de interfaz derivado del objeto `ifIndex` del grupo `interfaces`. Por ejemplo, la salida que aparece listada enseguida incluye el objeto `ipAdEntIf-Index.12.77.195.141`, con un valor de 3. El 3 representa la tercera interfaz en el sistema que resulta ser un enlace punto a punto (`ppp0`). La dirección IP `12.77.195.141` se asigna a esta interfaz. Podemos determinar el tipo de la tercera interfaz examinando el objeto `ifDescr` de la tabla `interfaces`. El objeto `ipAdEntNetMask` identifica la máscara de la red o, si es aplicable, la máscara de la subred para la dirección IP (`ipAdEntAddr`). El objeto `ipAdEntBcastAddr` muestra el valor del bit menos significativo de las direcciones de difusión IP empleadas para enviar paquetes a la interfaz lógica relacionada con esta dirección IP. El siguiente es un listado de ejemplo de la tabla `ipAddrTable`:

```
ip.ipAddrTable.ipAddrEntry.ipAdEntAddr.10.0.2.201 = IpAddress:
   10.0.2.201
ip.ipAddrTable.ipAddrEntry.ipAdEntAddr.12.77.195.141 = IpAddress:
   12.77.195.141
ip.ipAddrTable.ipAddrEntry.ipAdEntAddr.127.0.0.1 = IpAddress:
   127.0.0.1
ip.ipAddrTable.ipAddrEntry.ipAdEntIfIndex.10.0.2.201 = 2
ip.ipAddrTable.ipAddrEntry.ipAdEntIfIndex.12.77.195.141 = 3
ip.ipAddrTable.ipAddrEntry.ipAdEntIfIndex.127.0.0.1 = 1
ip.ipAddrTable.ipAddrEntry.ipAdEntNetMask.10.0.2.201 = IpAddress:
   255.255.255.0
ip.ipAddrTable.ipAddrEntry.ipAdEntNetMask.12.77.195.141 =
  IpAddress: 255.255.255.255
ip.ipAddrTable.ipAddrEntry.ipAdEntNetMask.127.0.0.1 = IpAddress:
   255.0.0.0
ip.ipAddrTable.ipAddrEntry.ipAdEntBcastAddr.10.0.2.201 = 1
ip.ipAddrTable.ipAddrEntry.ipAdEntBcastAddr.12.77.195.141 = 0
ip.ipAddrTable.ipAddrEntry.ipAdEntBcastAddr.127.0.0.1 = 1
ip.ipAddrTable.ipAddrEntry.ipAdEntReasmMaxSize.10.0.2.201 = -1
ip.ipAddrTable.ipAddrEntry.ipAdEntReasmMaxSize.12.77.195.141 = -1
ip.ipAddrTable.ipAddrEntry.ipAdEntReasmMaxSize.127.0.0.1 = -1
```

La tabla `ipRouteTable` proporciona información de enrutado IP para la entidad y se usa para tomar decisiones de enrutado cuando sea necesario. Para cada ruta conocida, hay una única entrada en la tabla, que es indexada por el objeto `ipRouteDest`. Para cada ruta, una interfaz local para el siguiente salto al destino está contenida en el objeto `ipRouteIfIndex`. Este valor coincide con el objeto `IfIndex` del grupo `interfaces`. El siguiente listado incluye información de enrutado relacionada con la red `199.70.195.0`:

```
ip.ipRouteTable.ipRouteEntry.ipRouteIfIndex.199.70.195.41 = 3
ip.ipRouteTable.ipRouteEntry.ipRouteMetric1.199.70.195.41 = 0
ip.ipRouteTable.ipRouteEntry.ipRouteMetric2.199.70.195.41 = -1
ip.ipRouteTable.ipRouteEntry.ipRouteMetric3.199.70.195.41 = -1
ip.ipRouteTable.ipRouteEntry.ipRouteMetric4.127.0.0.0 = -1
ip.ipRouteTable.ipRouteEntry.ipRouteMetric4.199.70.195.41 = -1
ip.ipRouteTable.ipRouteEntry.ipRouteNextHop.199.70.195.41 =
   IpAddress: 0.0.0.0
ip.ipRouteTable.ipRouteEntry.ipRouteType.199.70.195.41 = direct(3)
ip.ipRouteTable.ipRouteEntry.ipRouteProto.199.70.195.41 = local(2)
ip.ipRouteTable.ipRouteEntry.ipRouteAge.199.70.195.41 = 0
ip.ipRouteTable.ipRouteEntry.ipRouteMask.199.70.195.41 =
   IpAddress: 255.255.255.255
ip.ipRouteTable.ipRouteEntry.ipRouteMetric5.199.70.195.41 = -1
ip.ipRouteTable.ipRouteEntry.ipRouteInfo.199.70.195.41 = OID:
   .ccitt.nullOID
```

Téngase en cuenta que esta salida fue seleccionada de una lista de otras entradas, y formateada manualmente; la información de enrutado, normalmente no aparecería en el orden listado. La figura 6.7 presenta la vista de árbol de la tabla `ipRouteTable`. El objeto `ipRouteMetric1`, así como los otros tres objetos métricos se usan para denotar la medida de enrutado para esta ruta. Las definiciones actuales de la medida son determinadas por el protocolo de enrutado especificado por el valor `ipRouteProto`. En este caso, el valor cero indica que ésta es la ruta más directa para esta red. Cuando no se usa un objeto métrico particular (por ejemplo, `ipRouteMetric4.199.70.195.41`), el valor es `-1`.

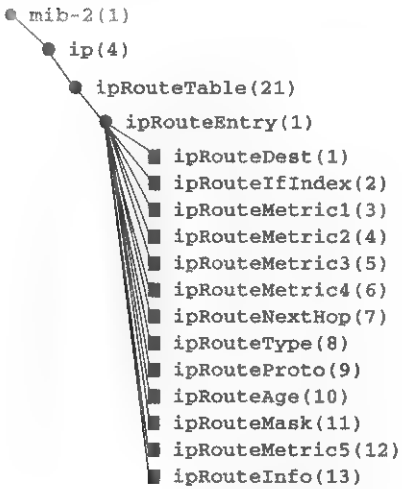


Figura 6.7 Vista de `ipRouteTable`.

El objeto `ipRouteNextHop` identifica el siguiente salto de la ruta. Un valor de `0.0.0.0` indica que esa dirección IP está limitada a la interfaz. El objeto `ipRouteType` indica el tipo de ruta e incluye los siguientes valores: `other(1)`, `invalid(2)`, `direct(3)` e `indirect(4)`. El objeto `ipRouteType` comunica la noción de enrutado directo e indirecto. El valor `invalid` invalida la ruta, y `other` significa que ninguno de los otros valores listados se aplica. El valor `direct` significa que la dirección de destino está en una subred conectada directamente, mientras que `indirect` significa que la dirección de destino no está en una red añadida pero, por lo menos, debe atravesarse una ruta adicional antes de poder llegar al destino. El objeto `ipRouteProto`

indica el mecanismo utilizado para aprender la ruta. La tabla 6.4 lista los valores `ipRouteProto` para un ejemplo en el cual el método empleado fue `local`. Este valor significa que la ruta fue configurada manualmente, sin la ayuda de ningún protocolo de enrutado. Esto se puede hacer, por ejemplo, cuando una interfaz es iniciada por primera vez.

Tabla 6.4 Valores **ipRouteProto** definidos.

Valor ipRouteProto	Descripción
`other`	Ninguno de los siguientes valores
`local`	Información de enrutado que se configura manualmente o se deriva de una fuente de protocolo de no enrutado
`netmgmt`	Información de enrutado obtenida de un protocolo de administración de red
`icmp`	Retroalimentación desde *hosts* y otros dispositivos acerca de un problema; suministra una redirección de mensaje que da información específica de enrutado
`egp`	Protocolo de puerta de enlace exterior (EGP), que está descrito en RFC904; es un protocolo simple de trabajo en red enrutador a enrutador
`ggp`	Protocolo de puerta de enlace a puerta de enlace; es un protocolo de enrutado utilizado por las primeras implementaciones de Internet, pero que ahora es obsoleto
`hello`	Un protocolo simple de enrutado que se emplea entre rutas contenidas en el mismo sistema autónomo

Tabla 6.4 Valores **ipRouteProto** definidos. (*continuación*)

Valor ipRouteProto	Descripción
rip	Protocolo de información de enrutado (*Routing Information Protocol,* RIP). Es el protocolo de enrutado de hecho para sistemas UNIX
is-is	Protocolo de sistema intermedio a sistema intermedio
es-is	Protocolo de sistema final a sistema intermedio
ciscolgrp	Protocolo de puerta de enlace interior propietario de Cisco
bbnSpfIgp	Protocolo de puerta de enlace interior de la ruta más corta primero, de BBN
ospf	RFC1583, protocolo de enrutado de puerta de enlace autónoma de ruta más corta abierta primero (OSPF)
bgp	RFC1771, protocolo de frontera de puerta de enlace (BGP). Protocolo autónomo de enrutado de puerta de enlace

El objeto `ipRouteAge` representa la cantidad de segundos transcurridos desde que esta ruta fue actualizada por última vez, o desde que se determinó que es una ruta correcta. El objeto `ipRouteMask` indica la máscara que va a ser operada mediante la operación lógica `y(&)` con la dirección de destino antes de ser comparada con el valor en el objeto `ipRouteDest`. El objeto `ipRouteMetric5` es una medida de ruta alterna para esta ruta y, como los demás, su valor es `-1` cuando no está en uso. El objeto final, `ipRouteInfo`, se usa como referencia de una cadena MIB OID que es específica del protocolo de enrutado particular responsable de esta ruta. Si la información no está disponible, debe usarse un identificador válido de objeto de `00`.

La tabla `ipRouteTable` tiene los siguientes objetos:

Nombre del objeto:	ipRouteEntry
OID:	ipRouteTable.1
Tipo de objeto:	Sequence
Modo de acceso:	na
Descripción:	Ruta a un destino particular

Nombre del objeto:	ipRouteDest
OID:	ipRouteEntry.1
Tipo de objeto:	ipAddress
Modo de acceso:	lectura-escritura
Descripción:	Dirección IP de destino para la definición de esta ruta

Nombre del objeto:	ipRouteIfIndex
OID:	ipRouteEntry.2
Tipo de objeto:	Integer
Modo de acceso:	lectura-escritura
Descripción:	Índice que identifica la interfaz local a través de la cual se puede llegar al siguiente salto de enrutado para esta ruta

Nombre del objeto:	ipRouteMetric1
OID:	ipRouteEntry.3
Tipo de objeto:	Integer
Modo de acceso:	lectura-escritura
Descripción:	Métrica de enrutado principal para esta ruta

Nombre del objeto:	ipRouteMetric2
OID:	ipRouteEntry.4

Tipo de objeto:	Integer
Modo de acceso:	lectura-escritura
Descripción:	Métrica alterna de enrutado para esta ruta

Nombre del objeto:	ipRouteMetric3
OID:	IpRouteEntry.5
Tipo de objeto:	Integer
Modo de acceso:	lectura-escritura
Descripción:	Métrica alterna de enrutado para esta ruta

Nombre del objeto:	ipRouteMetric4
OID:	IpRouteEntry.6
Tipo de objeto:	Integer
Modo de acceso:	lectura-escritura
Descripción:	Métrica alterna de enrutado para esta ruta

Nombre del objeto:	ipRouteNextHop
OID:	ipRouteEntry.7
Tipo de objeto:	IpAddress
Modo de acceso:	lectura-escritura
Descripción:	Dirección IP del próximo salto para esta ruta

Nombre del objeto:	ipRouteType
OID:	ipRouteEntry.8
Tipo de objeto:	Integer

Modo de acceso:	`lectura escritura`
Descripción:	Identificador de tipo para esta ruta. Los tipos válidos incluyen `other(1)`, `invalid(2)`, `direct(3)` e `indirect(4)`

Nombre del objeto:	`ipRouteProto`
OID:	`ipRouteEntry.9`
Tipo de objeto:	`Integer`
Modo de acceso:	`sólo lectura`
Descripción:	Mecanismo de enrutado que se emplea para aprender esta ruta

Nombre del objeto:	`ipRouteAge`
OID:	`ipRouteEntry.10`
Tipo de objeto:	`IpAddress`
Modo de acceso:	`lectura-escritura`
Descripción:	El tiempo transcurrido (en segundos) desde que la ruta fue actualizada o verificada

Nombre del objeto:	`ipRouteMask`
OID:	`ipRouteEntry.11`
Tipo de objeto:	`IpAddress`
Modo de acceso:	`lectura-escritura`
Descripción:	Dirección de máscara que se usa (operación lógica y) con la dirección de destino

Nombre del objeto:	`ipRouteMetric5`
OID:	`ipRouteEntry.12`

Tipo de objeto:	Integer
Modo de acceso:	lectura-escritura
Descripción:	Métrica alterna de enrutado para esta ruta

Nombre del objeto:	ipRouteInfo
OID:	ipRouteEntry.13
Tipo de objeto:	Object Identifier
Modo de acceso:	sólo lectura
Descripción:	Referencia de la definición MIB para el protocolo de enrutado responsable de esta ruta

La tabla ipNetToMediaTable es una tabla de traducción de direcciones que contiene entradas que corresponden a direcciones IP y a direcciones físicas. La figura 6.8 muestra la vista de árbol de esta tabla.

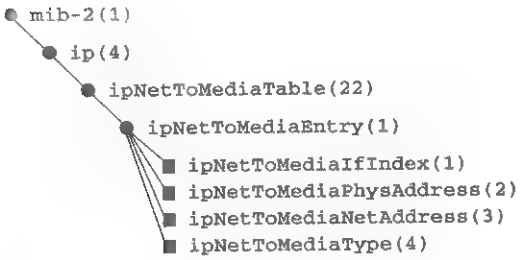


Figura 6.8 Vista de la tabla ipNetToMediaTable.

La información contenida en esta tabla es básicamente la misma que contiene la tabla at. La única diferencia es el objeto ipNetToMediaType que lista el tipo de transformación usado.

Nombre del objeto:	ipNetToMediaIfIndex
OID:	ipNetToMediaEntry.1

Tipo de objeto:	`Integer`
Modo de acceso:	`lectura-escritura`
Descripción:	Número de paquetes que fueron descartados a pesar de que eran válidos

La tabla `ipNetToMediaTable` contiene el siguiente grupo de objetos:

Nombre del objeto:	`ipNetToMediaPhysAddress`
OID:	`ipNetToMediaEntry.2`
Tipo de objeto:	`PhysAddress`
Modo de acceso:	`lectura-escritura`
Descripción:	Dirección física dependiente del medio; por ejemplo, una dirección Ethernet

Nombre del objeto:	`ipNetToMediaNetAddress`
OID:	`ipNetToMediaEntry.3`
Tipo de objeto:	`IpAddress`
Modo de acceso:	`lectura-escritura`
Descripción:	Dirección IP que corresponde a la dirección física dependiente del medio

Nombre del objeto:	`ipNetToMediaType`
OID:	`ipNetToMediaEntry.4`
Tipo de objeto:	`Counter`
Modo de acceso:	`lectura-escritura`
Descripción:	Tipo de transformación que generó la dirección. Los tipos incluyen `other(1)`, `invalid(2)`, `dynamic(3)` y `static(4)`

A continuación se muestra un listado de ejemplo de la tabla ipNetToMediaTable:

```
ip.ipNetToMediaTable.ipNetToMediaEntry.ipNetToMediaIfIndex.1.10.0.2.127
   1
ip.ipNetToMediaTable.ipNetToMediaEntry.ipNetToMediaPhysAddress.1.10.0.2.12
   7 = 0:60:97:e:a3:6
ip.ipNetToMediaTable.ipNetToMediaEntry.ipNetToMediaNetAddress.1.10.0.2.127
   = IpAddress: 10.0.2.127
ip.ipNetToMediaTable.ipNetToMediaEntry.ipNetToMediaType.1.10.0.2.127 =
   dynamic(3)
```

Al objeto ipNetToMediaType se le asigna el valor dynamic(3), y los valores adicionales pueden incluir other(1), invalid(2) y static(4). La asignación dynamic implica que la entrada fue creada por algún medio automático, utilizando tal vez el protocolo de configuración dinámica de *host* (DHCP) o algún otro mecanismo.

Grupo de reenvío IP

La tabla de reenvío IP se agregó a MIB-II a causa de los problemas con la tabla ipRouteTable. Con la tabla de reenvío IP pueden aparecer rutas múltiples a un solo destino, pero el acceso a estas entradas depende del mecanismo implementado para tener acceso a esta tabla. Desafortunadamente, no se ha estandarizado una manera de obtener esta información de enrutado dentro de SNMP. El grupo ipForward se introdujo con este propósito y está colocado al final de la tabla IP, como se muestra en la figura 6.9.

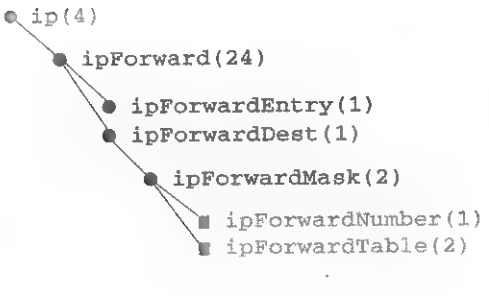


Figura 6.9 Vista de ipForward.

La mayor parte de los objetos de la tabla `IPForwardTable` corresponden a los objetos de la tabla `IPRouteTable` y tienen la misma sintaxis y significado. Como resultado, no se produjo un listado detallado de estos objetos. La principal diferencia estriba en que cada uno de los prefijos de los objetos fue cambiado de `ipRoute` a `ipForward`. Además, se adicionaron dos nuevos objetos: `ipForwardPolicy` e `ipForwardNextHopAs`. El objeto `ipForwardPolicy` indica la política de la ruta a un destino particular. Con el enrutado IP, la política se basa en el campo tipo de servicio IP contenido en el encabezado IP. Este campo define uno de los ocho niveles de precedencia y un valor binario de demora. El objeto `ipForwardNextHopAs` ayuda a quienes administran redes regionales suministrando el número del sistema del siguiente salto dentro de un dominio de enrutado autónomo.

El objeto `ipRouteDest` se usa como el índice en la tabla `ipRouteTable`. La tabla `ipForwardTable` puede indexarse por `ipForwardDest`, `ipForwardProto`, `ipForwardPolicy` e `ipForwardNextHop`.

Grupo ICMP

El grupo `icmp` contiene contadores que registran la operación del protocolo de mensajes de control de Internet (*Internet Control Message Protocol*). ICMP diagnostica problemas de conectividad y otros problemas de la capa de red. La famosa utilidad `ping` usa este protocolo. Una lista de los objetos del grupo `icmp` aparece en la figura 6.10.

El grupo `icmp` contiene el siguiente conjunto de objetos:

Nombre del objeto:	`icmpInMsgs`
OID:	`icmp.1`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de mensajes ICMP recibidos

Nombre del objeto:	`icmpInErrors`
OID:	`icmp.2`

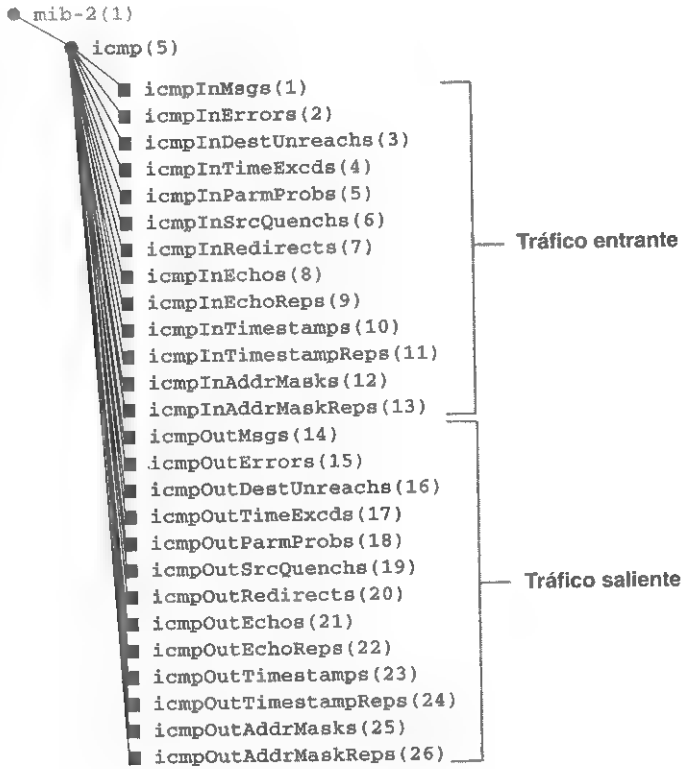


Figura 6.10 Vista del grupo `icmp`.

Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de mensajes ICMP recibidos que contenían algún error específico ICMP

Nombre del objeto:	`icmpInDestUnreachs`
OID:	`icmp.3`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de mensajes ICMP recibidos de destino no alcanzable

Nombre del objeto:	`icmpInTimeExcds`
OID:	`icmp.4`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de mensajes ICMP recibidos con tiempo excedido

Nombre del objeto:	`icmpInParmProbs`
OID:	`icmp.5`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de mensajes ICMP recibidos con problemas de parámetros

Nombre del objeto:	`icmpInSrcQuenchs`
OID:	`icmp.6`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de mensajes ICMP quench recibidos

Nombre del objeto:	`icmpInRedirects`
OID:	`icmp.7`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de mensajes ICMP de redirección recibidos

Nombre del objeto:	`icmpInEchos`
OID:	`icmp.8`

Tipo de objeto:	Counter
Modo de acceso:	sólo lectura
Descripción:	Número de mensajes ICMP de solicitud de eco recibidos

Nombre del objeto:	icmpEchoReps
OID:	icmp.9
Tipo de objeto:	Counter
Modo de acceso:	sólo lectura
Descripción:	Número de mensajes CIMP de respuesta de eco recibidos

Nombre del objeto:	icmpTimestamps
OID:	icmp.10
Tipo de objeto:	Counter
Modo de acceso:	sólo lectura
Descripción:	Número de mensajes ICMP de solicitud de sello de tiempo recibidos

Nombre del objeto:	icmpInTimestampReps
OID:	icmp.11
Tipo de objeto:	Counter
Modo de acceso:	sólo lectura
Descripción:	Número de mensajes ICMP de respuesta de sello de tiempo recibidos

Nombre del objeto:	icmpAddrMasks
OID:	icmp.12
Tipo de objeto:	Counter
Modo de acceso:	sólo lectura

Descripción: Número de solicitudes ICMP de máscara de dirección recibidas

Nombre del objeto: `icmpAddrMasksReps`
OID: `icmp.13`
Tipo de objeto: `Counter`
Modo de acceso: `sólo lectura`
Descripción: Número de mensajes ICMP de respuesta de máscara de dirección recibidos

Nombre del objeto: `icmpOutMsgs`
OID: `icmp.14`
Tipo de objeto: `Counter`
Modo de acceso: `sólo lectura`
Descripción: Número de mensajes ICMP que el sistema intentó enviar

Nombre del objeto: `icmpOutErrors`
OID: `icmp.15`
Tipo de objeto: `Counter`
Modo de acceso: `sólo lectura`
Descripción: Número de mensajes ICMP que el sistema no pudo enviar, debido a problemas ICMP

Nombre del objeto: `icmpOutDestUnreachs`
OID: `icmp.16`
Tipo de objeto: `Counter`
Modo de acceso: `sólo lectura`

Descripción:	Número de mensajes ICMP enviados con destino no alcanzable

Nombre del objeto:	`icmpOutTimeExcds`
OID:	`icmp.17`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de mensajes ICMP enviados con tiempo excedido

Nombre del objeto:	`icmpOutParmProbs`
OID:	`icmp.18`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de mensajes ICMP enviados con problemas de parámetros

Nombre del objeto:	`icmpOutSrcQuenchs`
OID:	`icmp.19`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de mensajes ICMP quench enviados

Nombre del objeto:	`icmpOutRedirects`
OID:	`icmp.20`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de mensajes ICMP de redirección enviados

Nombre del objeto:	`icmpOutEchos`
OID:	`icmp.21`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de paquetes que fueron descartados, a pesar de que eran válidos

Nombre del objeto:	`icmpOutEchoReps`
OID:	`icmp.22`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de mensajes ICMP de solicitud de eco enviados

Nombre del objeto:	`icmpOutTimestamps`
OID:	`icmp.23`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de mensajes ICMP de respuesta de eco enviados

Nombre del objeto:	`icmpOutTimestampReps`
OID:	`icmp.24`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de mensajes ICMP de solicitud de sello de tiempo enviados

Nombre del objeto:	icmpOutAddrMasks
OID:	icmp.25
Tipo de objeto:	Counter
Modo de acceso:	sólo lectura
Descripción:	Número de mensajes ICMP de respuesta de sello de tiempo enviados

Nombre del objeto:	icmpOutAddrMaskReps
OID:	icmp.26
Tipo de objeto:	Counter
Modo de acceso:	sólo lectura
Descripción:	Número de mensajes ICMP de solicitud de máscara de dirección enviados

Una muestra de la salida de una consulta al grupo icmp de un agente Linux incluye lo siguiente:

```
icmp.icmpInMsgs.0 = 27
icmp.icmpInErrors.0 = 0
icmp.icmpInDestUnreachs.0 = 7
icmp.icmpInTimeExcds.0 = 0
icmp.icmpInParmProbs.0 = 0
icmp.icmpInSrcQuenchs.0 = 0
icmp.icmpInRedirects.0 = 0
icmp.icmpInEchos.0 = 8
icmp.icmpInEchoReps.0 = 12
icmp.icmpInTimestamps.0 = 0
icmp.icmpInTimestampReps.0 = 0
icmp.icmpInaddrMasks.0 = 0
icmp.icmpInaddrMaskReps.0 = 0
icmp.icmpOutMsgs.0 = 10
icmp.icmpOutErrors.0 = 0
icmp.icmpOutDestUnreachs.0 = 2
icmp.icmpOutTimeExcds.0 = 0
icmp.icmpOutParmProbs.0 = 0
icmp.icmpOutSrcQuenchs.0 = 0
icmp.icmpOutRedirects.0 = 0
icmp.icmpOutEchos.0 = 0
icmp.icmpOutEchoReps.0 = 8
icmp.icmpOutTimestamps.0 = 0
icmp.icmpOutTimestampReps.0 = 0
icmp.icmpOutAddrMasks.0 = 0
icmp.icmpOutAddrMaskReps.0 = 0
```

Como se puede observar en la salida, muchos de los contadores son bastante bajos. Así es como debe ser. Los contadores altos para ciertos objetos, como `icmpInTimeExcds`, indican un problema de configuración u otro problema operacional.

Grupo TCP

El grupo `tcp` almacena información operacional y estadística acerca del protocolo de control de transmisión (TCP) en un sistema. Este grupo contiene una tabla sencilla que almacena información relacionada con cada conexión TCP en el sistema. La figura 6.11 muestra la vista en árbol de este grupo.

El grupo `tcp` tiene los siguientes objetos:

Nombre del objeto: `tcpRtoAlgorithm`

OID: `tcp.1`

Tipo de objeto: `Counter`

Modo de acceso: `sólo lectura`

Descripción: Algoritmo de tiempo de retransmisión; los valores posibles incluyen `other(1)`, `constant(2)`, `rsre(3)` y `vanj(4)`

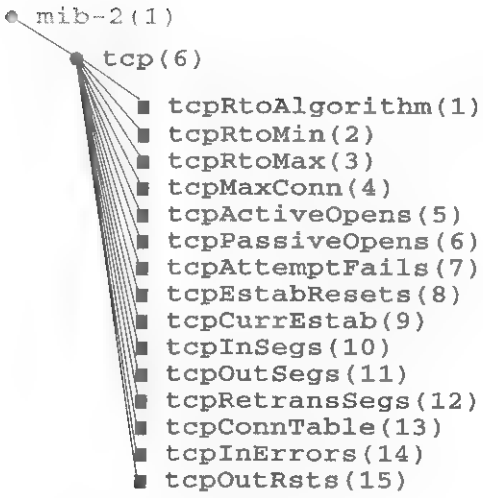


Figura 6.11 Vista del grupo `tcp`.

Nombre del objeto:	`tcpRtoMin`
OID:	`tcp.2`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Valor mínimo para el temporizador de retransmisión

Nombre del objeto:	`tcpRtoMax`
OID:	`tcp.3`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Valor máximo para el temporizador de retransmisión

Nombre del objeto:	`tcpMaxConn`
OID:	`tcp.4`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número total (o límite) de conexiones TCP que el sistema puede soportar

Nombre del objeto:	`tcpActiveOpens`
OID:	`tcp.5`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de conexión abiertas activas que este sistema ha soportado

Nombre del objeto:	`tcpPassiveOpens`
OID:	`tcp.6`

Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de conexiones abiertas pasivas que este sistema ha soportado

Nombre del objeto:	`tcpAttemptFails`
OID:	`tcp.7`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de intentos de conexiones fallidas que ocurrieron en este sistema

Nombre del objeto:	`tcpEstabResets`
OID:	`tcp.8`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de reinicios que han ocurrido en el sistema

Nombre del objeto:	`tcpCurrEstab`
OID:	`tcp.9`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de conexiones TCP que se encuentran en el estado *establecidas* (`ESTABLISHED`) o *cerradas en espera* (`CLOSE-WAIT`)

Nombre del objeto:	`tcpInSegs`
OID:	`tcp.10`

Tipo de objeto:	Counter
Modo de acceso:	sólo lectura
Descripción:	Número de segmentos recibidos, incluidos aquellos recibidos por error

Nombre del objeto:	tcpOutSegs
OID:	tcp.11
Tipo de objeto:	Counter
Modo de acceso:	sólo lectura
Descripción:	Número de segmentos enviados, exclusivos aquellos que contenían bytes retransmitidos

Nombre del objeto:	tcpRetransSegs
OID:	tcp.12
Tipo de objeto:	Counter
Modo de acceso:	sólo lectura
Descripción:	Número de segmentos retransmitidos

Nombre del objeto:	tcpConnTable
OID:	tcp.13
Tipo de objeto:	Counter
Modo de acceso:	sólo lectura
Descripción:	Información de conexión TCP

Nombre del objeto:	tcpInErrors
OID:	tcp.14
Tipo de objeto:	Counter
Modo de acceso:	sólo lectura
Descripción:	Número de segmentos recibidos por error

Nombre del objeto:	`tcpOutRsts`
OID:	`tcp.15`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de segmentos TCP enviados que contenían la bandera `RST`

Lo siguiente se obtuvo mediante sondeos a un sistema Linux:

```
tcp.tcpRtoAlgorithm.0   other(1)
tcp.tcpRtoMin.0 = 0
tcp.tcpRtoMax.0 = 0
tcp.tcpMaxConn.0 = 0
tcp.tcpActiveOpens.0 = 105422
tcp.tcpPassiveOpens.0 = 0
tcp.tcpAttemptFails.0 = 2968
tcp.tcpEstabResets.0   0
tcp.tcpCurrEstab.0 = Gauge: 2
tcp.tcpInSegs.0 = 63591
tcp.tcpOutSegs.0 = 111130
tcp.tcpRetransSegs.0 = 1881
```

La tabla `tcpConnTable` almacena información específica referente a las conexiones TCP en el sistema. La figura 6.12 muestra la presentación en árbol de esta tabla. El objeto `tcpConnEntry` se usa como un índice en la tabla. El tamaño de esta tabla se determina por el número de conexiones registradas en el sistema. En un servidor ocupado, el tamaño puede ser bastante grande.

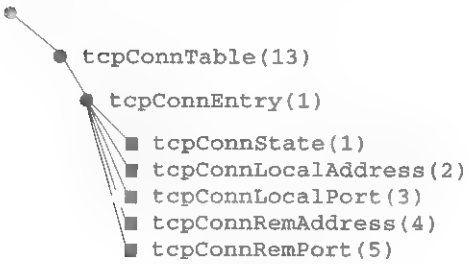


Figura 6.12 Vista de la tabla `tcpConnTable`.

El objeto `tcpConnState` identifica el estado presente de la conexión. Los valores válidos incluyen lo siguiente: `closed(1)`, `listen(2)`, `sysSent(3)`, `sysReceived(4)`, `established(5)`,

finWait1(6), finWait2(7), closeWait(8), lastAck(9), closing(10), timeWait(11) y deleteTCB(12). Excluido deleteTCB(12), los valores de estado son representativos del modelo de máquina de estados TCP que se analizó en el capítulo 2, "Protocolo simple de administración de red". Recuérdese que cada conexión deberá estar en uno de estos estados definidos. El estado deleteTCB(12) indica que el bloque de control TCP (TCB) para esta conexión se ha eliminado. Los objetos restantes almacenan tanto los pares dirección local y puerto como los pares dirección remota y puerto. Por ejemplo, considérese la siguiente salida:

```
tcp.tcpConnTable.tcpConnEntry.tcpConnState.10.0.2.201.23.10.0.2.127.1033 =
  timeWait(11)
tcp.tcpConnTable.tcpConnEntry.tcpConnState.10.0.2.201.23.10.0.2.127.1035 =
  established(5)
tcp.tcpConnTable.tcpConnEntry.tcpConnState.10.0.2.201.53.0.0.0.0.0 = listen(2)
tcp.tcpConnTable.tcpConnEntry tcpConnLocalAddress.10.0.2.201.23.10.0.2.127.1033=
  IpAddress: 10.0.2.201
tcp.tcpConnTable.tcpConnEntry.tcpConnLocalAddress.10.0.2.201.23.10.0.2.127.1035=
  IpAddress: 10.0.2.201
tcp.tcpConnTable.tcpConnEntry.tcpConnLocalAddress.10.0.2.201.53.0.0.0.0.0 =
  IpAddress: 10.0.2.201
tcp.tcpConnTable.tcpConnEntry.tcpConnLocalPort.10.0.2.201.23.10.0.2.127.1033
  = 23
tcp.tcpConnTable.tcpConnEntry.tcpConnLocalPort.10.0.2.201.23.10.0.2.127.1035
  = 23
tcp.tcpConnTable.tcpConnEntry.tcpConnLocalPort.10.0.2.201.53.0.0.0.0.0
  = 53
tcp.tcpConnTable.tcpConnEntry.tcpConnRemAddress.10.0.2.201.23.10.0.2.127.1033
  =IpAddress: 10.0.2.127
tcp.tcpConnTable.tcpConnEntry.tcpConnRemAddress.10.0.2.201.23.10.0.2.127.1035
  =IpAddress: 10.0.2.127 .
tcp.tcpConnTable.tcpConnEntry.tcpConnRemAddress.10.0.2.201.53.0.0.0.0.0 =
  IpAddress: 0.0.0.0
tcp.tcpConnTable.tcpConnEntry.tcpConnRemPort.10.0.2.201.23.10.0.2.127.1033
  = 1033
tcp.tcpConnTable.tcpConnEntry.tcpConnRemPort.10.0.2.201.23.10.0.2.127.1035
  = 1035
tcp.tcpConnTable.tcpConnEntry.tcpConnRemPort.10.0.2.201.53.0.0.0.0.0
  = 0
```

Dada la conexión

```
.tcpConnState.10.0.2.201.23.10.0.2.127.1035 = established(5)
```

que es el estado Established, el valor tcpConnLocalAddress es 10.0.2.201 y el valor de tcpConnLocalPort es 23, mientras la

información remota almacenada en `tcpConnRemoteAddress` y en `tcpConnRemotePort` es `10.0.2.201` y `1035`, respectivamente.

Grupo UDP

El grupo `udp` contiene objetos que se relacionan con la implementación y operación del protocolo de datagrama de usuario (UDP) en el sistema. Una tabla llamada `udpTable` registra la dirección local y la información de puerto para los servicios UDP a la escucha de solicitudes entrantes. Los objetos restantes son contadores que registran estadísticas específicas de tráfico UDP.

El grupo `udp` consta de los siguientes objetos:

Nombre del objeto:	`udpInDatagrams`
OID:	`udp.1`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de datagramas UDP entregados a protocolos y aplicaciones de mayor nivel

Nombre del objeto:	`udpNoPorts`
OID:	`udp.2`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de datagramas UDP recibidos, para los que no se ha proporcionado un puerto de aplicación específico

Nombre del objeto:	`udpInErrors`
OID:	`udp.3`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`

Descripción:	Número de datagramas UDP recibidos que no pudieron ser entregados por razones no relacionadas con tener una aplicación válida o un puerto de destino

Nombre del objeto:	`udpOutDatagrams`
OID:	`udp.4`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de datagramas UDP enviados desde este sistema

Nombre del objeto:	`udpTable`
OID:	`udp.5`
Tipo de objeto:	`Sequence of UdpEntry`
Modo de acceso:	`no accesible`
Descripción:	Información de la dirección y puerto del oyente UDP

Nombre del objeto:	`udpEntry`
OID:	`udpTable.1`
Tipo de objeto:	`Sequence`
Modo de acceso:	`no accesible`
Descripción:	`udpTable` para cada sesión UDP

Nombre del objeto:	`udpLocalAddress`
OID:	`udpEntry.1`
Tipo de objeto:	`pAddress`
Modo de acceso:	`sólo lectura`

Descripción:	Dirección local IP para este oyente, servicio o aplicación UDP
Nombre del objeto:	`udpLocalPort`
OID:	`udpEntry.2`
Tipo de objeto:	`Integer`
Modo de acceso:	`sólo lectura`
Descripción:	Número de puerto local (*socket*) para este oyente, servicio o aplicación UDP

La figura 6.13 muestra la vista en árbol del grupo udp. Los valores de ejemplo se obtuvieron mediante consultas al agente SNMP de Linux:

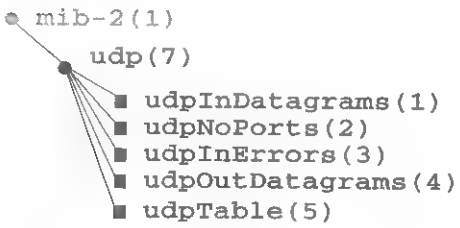


Figura 6.13 Vista del grupo udp.

```
udp.udpInDatagrams.0 = 1867
udp.udpNoPorts.0 = 8
udp.udpInErrors.0 = 0
udp.udpOutDatagrams.0 = 1951
udp.udpTable.udpEntry.udpLocalAddress.0.0.0.0.111 = IpAddress: 0.0.0.0
udp.udpTable.udpEntry.udpLocalAddress.0.0.0.0.161 = IpAddress: 0.0.0.0
udp.udpTable.udpEntry.udpLocalAddress.0.0.0.0.177 = IpAddress: 0.0.0.0
udp.udpTable.udpEntry.udpLocalAddress.0.0.0.0.517 = IpAddress: 0.0.0.0
udp.udpTable.udpEntry.udpLocalAddress.0.0.0.0.518 = IpAddress: 0.0.0.0
udp.udpTable.udpEntry.udpLocalAddress.0.0.0.0.615 = IpAddress: 0.0.0.0
udp.udpTable.udpEntry.udpLocalAddress.0.0.0.0.626 = IpAddress: 0.0.0.0
udp.udpTable.udpEntry.udpLocalAddress.0.0.0.0.636 = IpAddress: 0.0.0.0
udp.udpTable.udpEntry.udpLocalAddress.0.0.0.0.641 = IpAddress: 0.0.0.0
udp.udpTable.udpEntry.udpLocalAddress.0.0.0.0.646 = IpAddress: 0.0.0.0
udp.udpTable.udpEntry.udpLocalAddress.0.0.0.0.679 = IpAddress: 0.0.0.0
udp.udpTable.udpEntry.udpLocalAddress.0.0.0.0.800 = IpAddress: 0.0.0.0
```

(*continúa*)

```
udp.udpTable.udpEntry.udpLocalAddress.0.0.0.0.1022 = IpAddress: 0.0.0.0
udp.udpTable.udpEntry.udpLocalAddress.0.0.0.0.1023 = IpAddress: 0.0.0.0
udp.udpTable.udpEntry.udpLocalAddress.0.0.0.0.1024 = IpAddress: 0.0.0.0
udp.udpTable.udpEntry.udpLocalAddress.0.0.0.0.1026 = IpAddress: 0.0.0.0
udp.udpTable.udpEntry.udpLocalAddress.0.0.0.0.1041 = IpAddress: 0.0.0.0
udp.udpTable.udpEntry.udpLocalAddress.0.0.0.0.2049 = IpAddress: 0.0.0.0
udp.udpTable.udpEntry.udpLocalAddress.10.0.2.201.53 = IpAddress: 10.0.2.201
udp.udpTable.udpEntry.udpLocalAddress.127.0.0.1.53 = IpAddress: 127.0.0.1
udp.udpTable.udpEntry.udpLocalAddress.127.0.0.1.1119 = IpAddress: 127.0.0.1
udp.udpTable.udpEntry.udpLocalPort.0.0.0.0.111 = 111
udp.udpTable.udpEntry.udpLocalPort.0.0.0.0.161 = 161
udp.udpTable.udpEntry.udpLocalPort.0.0.0.0.177 = 177
udp.udpTable.udpEntry.udpLocalPort.0.0.0.0.517 = 517
udp.udpTable.udpEntry.udpLocalPort.0.0.0.0.518 = 518
udp.udpTable.udpEntry.udpLocalPort.0.0.0.0.615 = 615
udp.udpTable.udpEntry.udpLocalPort.0.0.0.0.626 = 626
udp.udpTable.udpEntry.udpLocalPort.0.0.0.0.636 = 636
udp.udpTable.udpEntry.udpLocalPort.0.0.0.0.641 = 641
udp.udpTable.udpEntry.udpLocalPort.0.0.0.0.646 = 646
udp.udpTable.udpEntry.udpLocalPort.0.0.0.0.679 = 679
udp.udpTable.udpEntry.udpLocalPort.0.0.0.0.800 = 800
udp.udpTable.udpEntry.udpLocalPort.0.0.0.0.1022 = 1022
udp.udpTable.udpEntry.udpLocalPort.0.0.0.0.1023 = 1023
udp.udpTable.udpEntry.udpLocalPort.0.0.0.0.1024 = 1024
udp.udpTable.udpEntry.udpLocalPort.0.0.0.0.1026 = 1026
udp.udpTable.udpEntry.udpLocalPort.0.0.0.0.1041 = 1041
udp.udpTable.udpEntry.udpLocalPort.0.0.0.0.2049 = 2049
udp.udpTable.udpEntry.udpLocalPort.10.0.2.201.53 = 53
udp.udpTable.udpEntry.udpLocalPort.127.0.0.1.53 = 53
udp.udpTable.udpEntry.udpLocalPort.127.0.0.1.1119 = 1119
```

Como la mayor parte de los sistemas UNIX, un conjunto central de servicios y aplicaciones para trabajo en red emplea una dirección estándar cuando se vincula a un puerto local. Se usa la dirección estándar `0.0.0.0`, y significa este *host*. Por ejemplo, en la salida de impresión anterior, el objeto

```
udpLocalAddress.0.0.0.0.161 = IpAddress: 0.0.0.0
```

demuestra que se usó una dirección local `0.0.0.0`, la cual está atada al puerto UDP `161`. El servicio que está asociado con este puerto es el proceso SNMP (o `snmpd`), y `udpLocalPort.0.0.0.0.161 = 161` (que es del tipo `integer`) retiene la información de este puerto. Como se puede observar, un gran número de servicios están corriendo en la tabla `udpEntry`, que contiene una entrada para cada servicio.

Si se quisiera determinar qué proceso está asociado con qué puerto, se puede buscar el número del puerto en el archivo `/etc/services`, o ejecutar el comando **netstat**. Considérese la siguiente salida del comando **netstat:**

```
# netstat -n -a -udp
(Not all processes could be identified, non-owned process info
will not be shown, you would have to be root to see it all.)
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local        Address Foreign Address State
   PID/Program name
udp        0      0 0.0.0.0:177     0.0.0.0:*   699/
udp        0      0 127.0.0.1:1119  0.0.0.0:*   652/actived
udp        0      0 0.0.0.0:800     0.0.0.0:*   -
udp        0      0 0.0.0.0:1022    0.0.0.0:*   501/amd
udp        0      0 0.0.0.0:679     0.0.0.0:*   501/amd
udp        0      0 0.0.0.0:1023    0.0.0.0:*   501/amd
udp        0      0 0.0.0.0:1026    0.0.0.0:*
udp        0      0 0.0.0.0:2049    0.0.0.0:*   -
udp        0      0 0.0.0.0:646     0.0.0.0:*   460/rpc.mountd
udp        0      0 0.0.0.0:641     0.0.0.0:*   460/rpc.mountd
udp        0      0 0.0.0.0:636     0.0.0.0:*   460/rpc.mountd
udp        0      0 0.0.0.0:626     0.0.0.0:*   449/rpc.rquotad
udp        0      0 0.0.0.0:615     0.0.0.0:*   438/rpc.statd
udp        0      0 0.0.0.0:161     0.0.0.0:*   376/snmpd
udp        0      0 0.0.0.0:1024    0.0.0.0:*   406/named
udp        0      0 10.0.2.201:53   0.0.0.0:*   406/named
udp        0      0 127.0.0.1:53    0.0.0.0:*   406/named
udp        0      0 0.0.0.0:518     0.0.0.0:*   362/inetd
udp        0      0 0.0.0.0:517     0.0.0.0:*   362/inetd
udp        0      0 0.0.0.0:111     0.0.0.0:*   255/portmap
```

El proceso `snmpd`, que está en negrilla, contiene la misma información de la tabla `udpTable`. *Véase* el capítulo 4, "Herramientas y utilidades centrales del sistema", para obtener una descripción completa del comando **netstat**.

Grupo EGP

El grupo `egp` contiene contadores y objetos relacionados con la operación e implementación del protocolo de puerta de enlace externo en un sistema. Como la mayor parte de los sistemas UNIX no implementan este grupo, no se incluyó su descripción. Sin embargo, es importante notar que algunos dispositivos de trabajo en red, como aquellos que suministran servicios de enrutado, soportarán a este grupo MIB.

Grupo de transmisión (dot3)

El grupo `dot3` contiene objetos que suministran detalles acerca del medio de enlace de datos de bajo nivel para cada una de las interfaces definidas en el sistema. Actualmente, el grupo `dot3` no es realmente un grupo en absoluto, sino más bien un nodo bajo la jerarquía de la rama MIB II. El propósito principal de este grupo es suministrar información específica respecto de una interfaz, en la forma de una MIB específica de interfaz. Recuérdese que el grupo `interfaces` suministra información genérica que se aplica a todas las interfaces en un sistema dado. Las MIB específicas de interfaces bajo `dot3` contienen información que está relacionada con un protocolo de enlace de datos específico, como Ethernet.

La MIB Ethernet, definida en RFC1643, define un conjunto de objetos que representan atributos de un medio de comunicación similar a Ethernet. La MIB EtherLike contiene tres métodos de acceso diferentes (o `ifType`), que son `ethernet-csmacd`, `iso88023-csmacd` y `starlan`. El método `ethernet-csmacd` es para el estándar Ethernet de operación sobre 10 Mbps, usando como medio cable coaxial de banda base. El método `iso88023 csmacd` cubre todos los estándares desarrollados por el comité IEEE 802.3 y el estándar internacional ISO 8802-3. Incluido entre la topología y los medios de cableada está Fast Ethernet, que utiliza UTP. El método `starlan` cubre la topología LAN obsoleta de 1Mbps. Nótese que solo tecnologías de trabajo en red como Ethernet usan esta MIB. Otros tipos de medios de trabajo en red, FDDI, por ejemplo, no tienen colisiones. Los dispositivos que soportan estos tipos de interfaces normalmente no soportan este grupo MIB.

La figura 6.14 presenta las cuatro tablas que incluye el grupo `dot3: dot3StateTable, dot3CollTable, dot3Tests` y `dot3StateErrors`. La tabla `dot3StateTable` contiene estadísticas sobre el tráfico de red obtenidas en la interfaz entre el agente de software y el medio de red. Se requiere una tabla, porque un dispositivo puede tener varias interfaces (por ejemplo, un enrutador multiprotocolo).

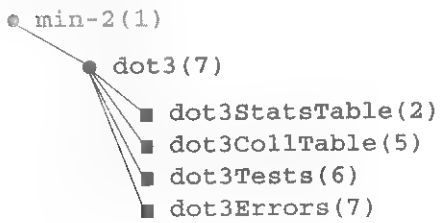


Figura 6.14 Vista del grupo de transmisión `dot3`.

La tabla `dot3StateTable` registra estadísticas respecto al tráfico observado entre el agente y el medio de red físico. Es necesaria una tabla, en lugar de un conjunto de contadores específicos, porque las estaciones de trabajo, servicios y otros dispositivos de trabajo en red pueden contener más de una interfaz de red. Por tanto, hay una entrada en la tabla para cada interfaz definida en el sistema. Esta tabla es indexada por el objeto `dot3StateIndex`. Los objetos de la tabla `dot3StateTable` aparecen en la figura 6.15.

La `dot3StateTable` contiene los siguientes objetos:

Nombre del objeto: `dot3StateEntry`

OID: `dot3StateTable.1`

Tipo de objeto: `Sequence`

Modo de acceso: `sólo lectura`

Descripción: Estadísticas para una interfaz de red similar a Ethernet

Figura 6.15 Vista de la tabla `dot3StateTable`.

Nombre del objeto: `dot3StateIndex`

OID: `dot3StateEntry.1`

Tipo de objeto: `Integer`

Modo de acceso:	sólo lectura
Descripción:	Identifica una interfaz en la tabla dot3StateTable

Nombre del objeto:	dot3StateAlignmentErrors
OID:	dot3StateEntry.2
Tipo de objeto:	Counter
Modo de acceso:	sólo lectura
Descripción:	Marcos leídos desde la red que está contenida dentro de los límites de bytes correctos

Nombre del objeto:	dot3StateFCSErrors
OID:	dot3StateEntry.3
Tipo de objeto:	Counter
Modo de acceso:	sólo lectura
Descripción:	Marcos leídos desde la red, que no pasaron la secuencia de verificación de marcos (FCS)

Nombre del objeto:	dot3StateSingleCollisionFrames
OID:	dot3StateEntry.4
Tipo de objeto:	Counter
Modo de acceso:	sólo lectura
Descripción:	Marcos transmitidos en la red que causaron exàctamente una colisión

Nombre del objeto:	dot3StateMultipleCollisionFrames
OID:	dot3StateEntry.5
Tipo de objeto:	Counter
Modo de acceso:	sólo lectura

Descripción:	Marcos transmitidos en la red que causaron más de una colisión
Nombre del objeto:	`dot3StateSQETestErrors`
OID:	`dot3StateEntry.6`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Frecuencia para la generación de la revisión de prueba SQE
Nombre del objeto:	`dot3StateDeferredTransmission`
OID:	`dot3StateEntry.7`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de marcos para los cuales se demoró el primer intento de transmisión, debido a contención en la red
Nombre del objeto:	`dot3StateLateCollisions`
OID:	`dot3StateEntry.8`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de veces que se detectó una colisión, a intervalos de tiempo de 512 bits mientras se transmitía
Nombre del objeto:	`dot3StateExcessive Collisions`
OID:	`dot3StateEntry.9`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`

Descripción:	Número de marcos transmitidos que fallaron, debido a colisiones excesivas

Nombre del objeto:	`dot3StateInternalMacTransmitErrors`
OID:	`dot3StateEntry.10`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de marcos transmitidos que fallaron, debido a errores de control de acceso a medios internos (MAC)

Nombre del objeto:	`dot3StateCarrierSenseErrors`
OID:	`dot3StateEntry.11`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de veces que la condición del *carrier sense* (CS) estuvo perdida o no asegurada cuando se transmitía un marco

Nombre del objeto:	`dot3StateFrameTooLong`
OID:	`dot3StateEntry.13`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de marcos recibidos que excedieron el tamaño máximo de marco

Nombre del objeto:	`dot3StateInternalMacReceiveErrors`
OID:	`dot3StateEntry.16`
Tipo de objeto:	`Counter`

Modo de acceso:	`sólo lectura`
Descripción:	Número de marcos recibidos que fallaron, debido a errores de control de acceso al medio interno (MAC)

Nombre del objeto:	`dot3StateEtherChipSet`
OID:	`dot3StateEntry.17`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número del chipset asociado con esta interfaz

La tabla `dot3CollTable` contiene un registro de la actividad de colisión observada en la red entre el agente y el medio de la red. Almacena un histograma para un conjunto de interfaces definidas en el sistema. El objeto `dot3CollCount` contiene un único punto de histograma para cada interfaz. El objeto `dot3CollFrequencies` almacena el número de marcos transmitidos que experimentaron exactamente el número de colisiones en el objeto asociado `dot3CollCount`.

La tabla `dot3CollTable` contiene los siguientes objetos:

Nombre del objeto:	`dot3CollEntry`
OID:	`dot3CollTable.1`
Tipo de objeto:	`Sequence`
Modo de acceso:	`sólo lectura`
Descripción:	Grupo de histogramas de colisión para una colección de interfaces de red

Nombre del objeto:	`dot3CollCount`
OID:	`dot3StateEntry.1`
Tipo de objeto:	`Counter`

Modo de acceso:	sólo lectura
Descripción:	Punto de histograma para una interfaz específica. Número total de colisiones para una interfaz específica

Nombre del objeto:	dot3CollFrequencies
OID:	dot3StateEntry.2
Tipo de objeto:	Counter
Modo de acceso:	sólo lectura
Descripción:	Número de marcos de una interfaz que experimentó exactamente el número de colisiones en el objeto asociado dot3CollCount

Las dos tablas restantes, dot3Tests y dot3Errors, se usan cuando se prueban interfaces y cuando se reportan errores. La tabla dot3Tests contiene dos objetos, dot3TestTdr y dot3Test_oopBack, que identifican pruebas específicas que pueden invocarse en los medios de la red. El objeto dot3TestTdr invoca una prueba de reflectometría en el dominio del tiempo (TDR) mientras que el objeto dot3Test_oopBack suministra una prueba de ciclo de retorno del hardware. La tabla dot3Errors también contiene dos objetos: dot3ErrorInitError y dot3ErrorLoopBack«Error, que almacenan información de errores relacionados con cualesquiera pruebas que se puedan haber ejecutado desde la tabla dot3Tests.

Las tablas dot3Tests y dot3Errors contienen los siguientes objetos:

Nombre del objeto:	dot3TestsTDR
OID:	dot3Tests.1
Tipo de objeto:	Counter
Modo de acceso:	sólo lectura
Descripción:	Prueba TDR

Nombre del objeto:	`dot3TestLoopBack`
OID:	`dot3Tests.2`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Prueba de ciclo de retorno (*Loopback test*)

Nombre del objeto:	`dot3ErrorInitError`
OID:	`dot3Tests.1`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Identifica errores relacionados con el chipset que falló las pruebas

Nombre del objeto:	`dot3StateIndex`
OID:	`dot3StateEntry.1`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Errores relacionados con la ejecución de las pruebas de ciclo de retorno

Grupo SNMP

El grupo `snmp` contiene información relacionada con la operación e implementación de SNMP. Algunos de los objetos de este grupo están relacionados con la administración de la estación SNMP o con las funciones del agente. Por tanto, cada implementación SNMP puede usar ciertos objetos de la tabla. Los objetos que no están soportados por el agente, contienen un valor de cero.

Con la excepción del objeto `snmpEnableAuthenTraps`, que es de *lectura-escritura*, todos los demás objetos son de *sólo lectura*. Este objeto controla si al agente se le permite generar trampas de falla-autenticación, y generalmente se coloca por una aplicación de administración de la red. Las fallas de autenticación son el

resultado del intento de una entidad administrativa de ejecutar alguna función privilegiada en el agente, cuando no cuenta con el permiso de seguridad correcto, o con credenciales para hacerlo. La figura 6.16 muestra la vista en árbol de la tabla snmp.

Figura 6.16 Vista del grupo snmp.

Nótese que ambas posiciones, 7 y 8, no aparecen listadas porque la MIB no define ningún objeto para estos espacios. Como se puede observar en la figura 6.16, muchos de los objetos SNMP aparecen en color gris en la lista, mientras que unos cuantos aparecen en negro. Esto se debe a que la versión original de la tabla incluía todos los objetos grises, y también algunos de los otros. Cuando se publicó la especificación SNMPv2, muchos de estos objetos fueron depreciados (abandonados). Sin embargo, como muchos agentes continúan soportando a SNMPv1, todavía se

usan estos objetos. Éste es un ejemplo de cuándo las especificaciones MIB nuevas tienen poco efecto sobre lo que se implementó en el pasado. Los objetos más antiguos del grupo `snmp` nos acompañarán por largo tiempo. La tabla `snmp` contiene los siguientes objetos:

Nombre del objeto:	`snmpInPkts`
OID:	`snmp.1`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de mensajes SNMP entregados al agente

Nombre del objeto:	`snmpOutPkts` (**depreciado**)
OID:	`snmp.2`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de mensajes SNMP pasados del agente a la capa de transporte

Nombre del objeto:	`snmpInBadVersions`
OID:	`snmp.3`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de mensajes SNMP pasados al agente y que no fueron soportados

Nombre del objeto:	`snmpInBadCommunityNames`
OID:	`snmp.4`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`

Descripción:	Número de mensajes SNMP pasados al agente y que contenían el nombre de una comunidad desconocida

Nombre del objeto:	`snmpInBadCommunityUses`
OID:	`snmp.5`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de mensajes SNMP pasados al agente y que contenían una operación SNMP no permitida por el nombre de comunidad SNMP suministrado

Nombre del objeto:	`snmpInASNParseErrs`
OID:	`snmp.6`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	El número de errores BER y ASN encontrados cuando se decodificaron los mensajes SNMP entrantes

Nombre del objeto:	`snmpInTooBigs` **(depreciado)**
OID:	`snmp.8`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de PDU SNMP entregados al agente, para las cuales ocurrió el error `tooBig`

Nombre del objeto:	`snmpInNoSuchName` **(depreciado)**
OID:	`snmp.9`

Tipo de objeto: Counter
Modo de acceso: sólo lectura
Descripción: Número de PDU SNMP entregados al agente, para las cuales ocurrió el error noSuchName

Nombre del objeto: snmpInBadValues (depreciado)
OID: snmp.10
Tipo de objeto: Counter
Modo de acceso: sólo lectura
Descripción: El número de PDU SNMP entregados al agente, para las cuales ocurrió el error badValue

Nombre del objeto: snmpInReadOnlys (depreciado)
OID: snmp.11
Tipo de objeto: Counter
Modo de acceso: sólo lectura
Descripción: Número de PDU SNMP entregados al agente, para las cuales ocurrió el error readOnly

Nombre del objeto: snmpInGenErrs (depreciado)
OID: snmp.12
Tipo de objeto: Counter
Modo de acceso: sólo lectura
Descripción: Número de PDU SNMP entregados al agente, para las cuales ocurrió el error genErr

Nombre del objeto: snmpInTotalReqVars (depreciado)
OID: snmp.13

Tipo de objeto:	Counter
Modo de acceso:	sólo lectura
Descripción:	Número de objetos MIB recuperados exitosamente por el agente, como resultado de una solicitud válida get-request SNMP y una solicitud getnext-request SNMP

Nombre del objeto:	snmpInTotalSetVars **(depreciado)**
OID:	snmp.14
Tipo de objeto:	Counter
Modo de acceso:	sólo lectura
Descripción:	Número de objetos MIB recuperados exitosamente por el agente, como resultado de una solicitud válida set-request SNMP

Nombre del objeto:	snmpInGetRequests **(depreciado)**
OID:	snmp.15
Tipo de objeto:	Counter
Modo de acceso:	sólo lectura
Descripción:	El número de solicitudes get-requests SNMP aceptadas y procesadas por el agente

Nombre del objeto:	snmpInGetNexts **(depreciado)**
OID:	snmp.16
Tipo de objeto:	Counter
Modo de acceso:	sólo lectura
Descripción:	El número de solicitudes getnext-requests SNMP aceptadas y procesadas por el agente

Nombre del objeto:	`snmpInSetRequests`
OID:	`snmp.17`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	El número de solicitudes `set-requests` SNMP aceptadas y procesadas por el agente

Nombre del objeto:	`snmpInGetResponses` **(depreciado)**
OID:	`snmp.18`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de `get-responses` SNMP aceptadas y procesadas por el agente

Nombre del objeto:	`snmpInTraps` **(depreciado)**
OID:	`snmp.19`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de `traps` SNMP aceptadas y procesadas por el agente

Nombre del objeto:	`snmpOutTooBig`
OID:	`snmp.20`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de PDU SNMP generadas por el agente, para las cuales ocurrió el error `TooBig`

Nombre del objeto: snmpOutNoSuchNames **(depreciado)**
OID: snmp.21
Tipo de objeto: Counter
Modo de acceso: sólo lectura
Descripción: Número de PDU SNMP generadas por el agente, para las cuales ocurrió el error noSuchName

Nombre del objeto: snmpOutBadValues **(depreciado)**
OID: snmp.22
Tipo de objeto: Counter
Modo de acceso: sólo lectura
Descripción: Número de PDU SNMP generadas por el agente, para las cuales ocurrió el error badValue

Nombre del objeto: snmpOutGenErrs **(depreciado)**
OID: snmp.24
Tipo de objeto: Counter
Modo de acceso: sólo lectura
Descripción: Número de PDU SNMP generadas por el agente, para las cuales ocurrió el error genErr

Nombre del objeto: snmpOutGetRequest **(depreciado)**
OID: snmp.25
Tipo de objeto: Counter
Modo de acceso: sólo lectura
Descripción: Número de PDU SNMP get-request generadas por el agente

Nombre del objeto:	`snmpOutGetNexts` **(depreciado)**
OID:	`snmp.26`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de PDU SNMP `getnext-request`, generadas por el agente

Nombre del objeto:	`snmpOutSetRequests` **(depreciado)**
OID:	`snmp.27`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de PDU SNMP `set-request` generadas por el agente

Nombre del objeto:	`snmpOutGetResponses` **(depreciado)**
OID:	`snmp.28`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de PDU SNMP `get-responses` generadas por el agente

Nombre del objeto:	`snmpOutTraps` **(depreciado)**
OID:	`snmp.29`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número de `traps` SNMP generadas por el agente

Nombre del objeto:	`snmpEnableAuthenTraps`
OID:	`snmp.30`

Tipo de objeto:	`Counter`
Modo de acceso:	`lectura-escritura`
Descripción:	Determina si deben generarse las trampas de fallas de autenticación. Los valores válidos son `enabled(1)` o `disabled(2)`. Cuando una entidad SNMP intente consultar al agente con una cadena de comunidad inválida o con un símbolo de autenticación, el agente emitirá una trampa, si está habilitada

07

Uso de los agentes SNMP

Este capítulo ofrece una vista detallada de los agentes SNMP que están disponibles específicamente en los sistemas Solaris y Linux. Los sistemas Red Hat Linux son equipados con el agente del sistema SNMP de la Universidad de California en Davis (UCD), que proporciona tanto objetos específicos MIB-II como UCD. El agente Sun, conocido como el Agente Empresarial Solstice (SEA), viene con la más reciente versión de Solaris e incluye objetos MIB específicos de Sun. El paquete SEA incluye, actualmente, dos agentes: un agente maestro y un subagente. Debido a que el término Solstice se refiere también a un marco de trabajo de administración de red de Sun y a una mezcla de productos, este libro utilizará los términos *agente maestro* y *subagente* para describir estos agentes de Sun.

Vista general de los agentes

Los tres agentes listados anteriormente suministran algunos de los mismos servicios fundamentales, a saber, la recuperación y colocación de objetos MIB. Cada uno de los agentes soporta las operaciones básicas estándar SNMP, incluidos `get`, `set` y `trap`. Adicionalmente, cada uno de los agentes proporciona algunas funciones especializadas como la administración de memoria o soporte para disco. La escogencia del agente que se va a utilizar depende del entorno y de los tipos de problemas que se intente resolver. Por ejemplo, puede ser necesario utilizar los tres agentes, porque cada uno proporciona servicios que se requieren.

La funcionalidad de cada uno de los agentes puede ser resumida en varias categorías:

Soporte MIB-II:	El agente soporta la mayor parte o todo el conjunto estándar de objetos MIB-II. El estándar MIB-II se describió en el capítulo anterior.
Administración del proceso:	Objetos adicionales MIB están disponibles para administrar procesos UNIX. Las operaciones básicas incluyen listado y control de estos procesos a través del agente.
Configuración remota:	El agente puede manipular su propio archivo de configuración a través del protocolo SNMP. Por ejemplo, el agente puede ser diseñado para alterar, dinámicamente, una acción predefinida, cuando el objeto seguido alcanza un umbral crítico.
Soporte MIB extendido:	El agente soporta objetos MIB adicionales, los cuales proporcionan información más allá del estándar MIB-II. Ésta puede incluir ya sea MIB propietarias específica del vendedor o MIB estándar, como la MIB Recurso de *host*.
Configuración flexible:	El agente proporciona un conjunto flexible de opciones de configuración

	y palabras clave, las cuales se usan principalmente para controlar el estado operativo y el comportamiento del agente.
Objetos de rendimiento:	El agente soporta objetos adicionales MIB que proporcionan información adicional de rendimiento o datos estadísticos.
Acciones personalizables:	El agente puede ejecutar secuencias de comandos o programas externos contenidos en el archivo de configuración del agente. Estas acciones son usualmente el resultado directo de alguna condición o evento específico que el agente ha detectado.
Administración de memoria/disco:	El agente proporciona objetos MIB adicionales que suministran información adicional del disco o de la memoria o datos estadísticos.
Soporte de agente maestro:	El agente proporciona la capacidad a los subagentes de registrar mensajes SNMP recibidos. El agente maestro es responsable de manejar la comunicación externa en nombre de los subagentes. El agente maestro utiliza el puerto 161 SNMP, mientras que los subagentes se comunican con el agente maestro en otro puerto predefinido.
Extensible:	El agente puede extenderse para soportar funcionalidad adicional y proporcionar nuevas características, como ejecutar las secuencias de comandos externas shell UNIX u otros comandos. Esto también puede significar que el agente suministra un entorno de desarrollo que incluye interfaces que facilitan la adicción de nuevas características

	por medio de la recopilación (o encadenamiento) de varios componentes del paquete.
SNMPv2/v3:	El agente soporta uno o más elementos de las especificaciones del protocolo SNMPv2 y/o del SNMPv3.

La tabla 7.1 resume la funcionalidad de cada agente y puede usarse como una referencia rápida, cuando se escoge un agente.

Tabla 7.1 Vista general funcional de los agentes.

Categoría	Linux	Agente Maestro Sun	Sun
Objetos MIB-II	✔		✔
Objetos de administración de procesos	✔	✔	✔
Configuración remota	✔		
Objetos adicionales del sistema			✔
Configuración flexible	✔		✔
Objetos adicionales de rendimiento	✔		✔
Acciones personalizables	✔		
Objetos de administración de memoria	✔		
Objetos de administración de disco	✔		
Agente maestro		✔	
Administración de subagente		✔	
Capacidades extensibles	✔		
SNMPv2/v3	✔		

El agente SNMP de Linux

El agente Linux, conocido como `snmpd`, suministra una lista detallada de los servicios relacionados con SNMP. El agente Linux es actualmente el agente de la Universidad de California en Davis conocido como el agente UCD, y está disponible en una

gran cantidad de sistemas operativos, incluido Linux. Este agente soporta los protocolos de administración de red SNMPv1, SNMPv2 y SNMPv3. Esto significa que no sólo soporta las operaciones básicas SNMP, sino también aquellas asociadas con el SNMPv2, como `get bulk`, `get-inform`, y otras. Además, soporta el Modelo de Seguridad del Usuario (USM), que se describe en las especificaciones del SNMPv3. Éste soporta MIB-II, MIB UCD, y otras MIB. El agente proporciona la capacidad de ejecutar de manera remota comandos UNIX configurados con base en ciertas condiciones o eventos. El agente puede usarse para hacer un seguimiento tanto de la memoria del sistema como de la información de procesos del sistema.

Las ventajas de usar el paquete UCD son significativas. Primero, el software suministra un conjunto muy completo de herramientas y características que son de dominio público. El paquete incluye herramientas del usuario SNMP, más un agente SNMP completamente funcional. Muchos paquetes SNMP están disponibles en Internet, pero no se comparan con la estabilidad, la robustez y las características que se encuentran en el paquete UCD. Segundo, no obstante que el software UCD es de dominio público, parece tener seguidores fieles y ha tenido mejoras continuas, aun superando productos comerciales bien establecidos en términos de soporte y características del protocolo SNMP. Las herramientas UCD están soportadas en un gran número de plataformas de computadora y sistemas operativos. El paquete se incluye en ciertas distribuciones de Linux, como Red Hat. Finalmente, el paquete UCD es de libre distribución, lo que significa que se puede instalar a través de la red empresarial sin preocuparse por asuntos de licencia; esto lo hace bastante atractivo para instituciones y otras organizaciones con presupuesto restringido.

Agente maestro de Sun

Sun proporciona un agente maestro estándar de administración de red SNMP que puede usarse como base de la estrategia de administración empresarial. Este agente, llamado `snmpdx`, proporciona una facilidad de agente maestro/subagente para uno o más subagentes. Recuérdese que en el capítulo 2, "Protocolo simple de administración de red", se expresó que el propósito principal y el papel de un agente maestro es permitir la coexistencia de muchos agentes SNMP. Debido a que cada agente SNMP puede escuchar las solicitudes en el bien conocido puerto

SNMP 161, debe proporcionarse un mecanismo, de modo que todos los agentes puedan compartir este puerto sin conflicto. Sin un arbitraje especial, el primer agente que emplea el puerto obtendrá un uso exclusivo del puerto, y todos los otros agentes tendrán acceso denegado. Con el agente `snmpdx`, pueden instalarse muchos subagentes en el mismo sistema, sin el problema tradicional de contención de puerto. Además, este agente maestro soporta una colección de objetos MIB que describen cada uno de los subagentes configurados con el agente maestro.

Agente SNMP de Sun

Sun también proporciona un agente estándar SNMP del sistema de administración de red. Este agente, conocido como `mibiisa`, soporta el SNMPv1 y proporciona acceso de lectura-escritura, a objetos MIB-II y a los objetos específicos Sun *enterprise*. Este agente trabaja bajo la arquitectura de agente maestro/esclavo de Sun, lo cual significa que puede operar en conjunto con el agente maestro o en un modo independiente. Suministra la capacidad de administrar estaciones de trabajo Sun y servidores desde un punto de vista operacional y de rendimiento. Éste soporta grupos de objeto MIB-II estándar y algunos grupos adicionales definidos por Sun. Juntas, las características de este agente pueden hacer un seguimiento a los problemas más críticos de la administración en red, que enfrentan los administradores de sistemas y de redes, por igual.

Las ventajas de usar el agente UCD incluyen la integración dentro del sistema operativo y soporte por parte del vendedor. Debido a que el agente es despachado con Solaris, el componente del agente puede cargarse muy fácilmente con el software del sistema operativo. En versiones anteriores del sistema operativo que no tienen todavía el agente, es posible instalar éste de una manera sencilla empleando la utilidad **pkgadd** de Sun. Además, debido a que el agente fue desarrollado por Sun, éste recibirá soporte a medida que surjan problemas que requieran asistencia técnica. Finalmente, el respaldo del vendedor asegura que los usuarios pueden esperar la inclusión de mejoras y nueva funcionalidad en futuras versiones del agente. Esto es importante porque SNMP continuará cambiando para satisfacer los crecientes requerimientos de los administradores del sistemas y de red.

Las MIB del agente

Cada uno de los agentes descritos en esta sección implementa tanto las MIB estándar como las privadas (o ambas). Debido a

que tanto los agentes Sun como los UCD utilizan MIB privadas, el acceso a estos objetos se da en la rama empresarial. La figura 7.1 muestra la ruta desde la raíz hasta cada una de las ramas privadas del vendedor.

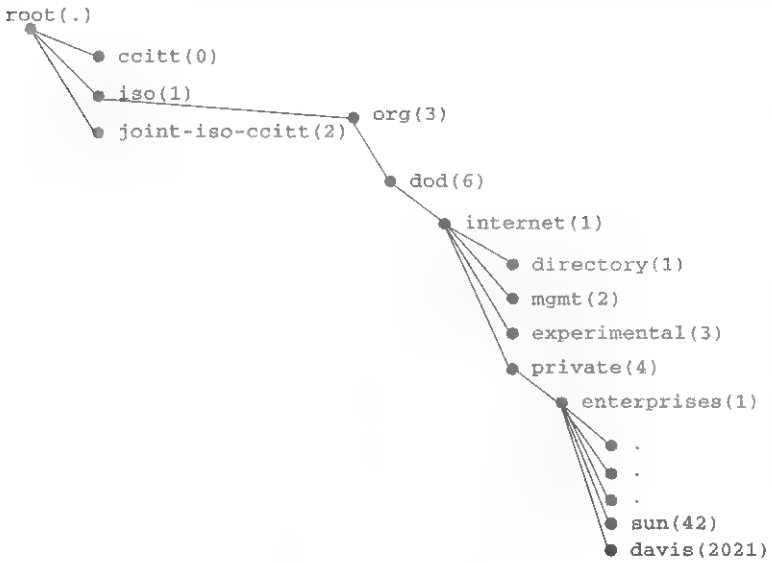


Figura 7.1 Ramas Sun/UCD del agente MIB.

Consideraciones de seguridad

Estos agentes son muy poderosos y pueden tener un impacto profundo en la administración de red. Debido a que suministran acceso a porciones críticas del sistema, debe tenerse cuidado cuando se configuran y se utilizan. Por ejemplo, el archivo de configuración para el Agente Sun proporciona la capacidad de establecer cadenas de comunidad tanto para el acceso de lectura como para el de escritura, a la MIB. Muchos sistemas de administración de red y agentes (incluidos los que se listaron aquí) están configurados de manera predeterminada con las cadenas de comunidad estándar `public` y `private`. Sería importante cambiar éstas a contraseñas no estándar cuando estos agentes se instalan en un sitio local. No obstante que la información de la cadena de comunidad SNMP puede obtenerse a través del dispositivo de captura de paquete y decodificarse, los nombres de comunidad deberían, aun, ser únicos y difíciles de descifrar. En una gran instalación, puede ser necesario usar un conjunto de cadenas de

comunidad para un grupo de dispositivos, ya sean colocalizadas geográficamente o de ciertos tipos, como los dispositivos Cisco o 3Com.

Fallar al configurar los agentes correctamente puede ser desastroso. Por ejemplo, con el agente Sun, una interfaz de red puede desactivarse con una simple operación `set` SNMP si la cadena de comunidad es `public`. Alguien, con un explorador MIB o con una herramienta de comando **snmpset**, puede desactivar fácilmente una o más interfaces del sistema. Si la interfaz desactivada es la única interfaz en el sistema, éste llegaría a ser desconectado de la red de forma efectiva. Como resultado, todos los procesos relacionados con la red se verían afectados. En algunos entornos, esto sería considerado sabotaje y acarreará consecuencias significativas. Para restaurar el sistema a su estado original, el administrador debe entrar en la consola del sistema, si está disponible, y activar la interfaz utilizando el comando **ifconfig.** De manera alternativa, si no existe consola del sistema, éste debe ser reiniciado apagándolo; este paso también incrementa la probabilidad de pérdida de datos, porque no hay forma de sincronizar, de antemano, los sistemas de archivos.

La regla general en este caso es tener cuidado cuando se instalan los archivos de configuración de agente y no usar los nombres de comunidad estándar SNMP. Como se verá, una lista de control de acceso (ACL) y facilidades de seguridad adicionales disponibles en los agentes proporcionarán mayor protección.

El agente Linux SNMP

Descripción general

El Agente Linux (`snmpd`) es un agente del sistema SNMP muy poderoso, extensible y completamente funcional, que está disponible en una gran cantidad de plataformas UNIX. Para una lista de plataformas específicas soportadas, *véase* la sección de instalación en el apéndice B. Como se mencionó con anterioridad, el agente soporta MIB-II (estadísticas de red), MIB SNMPv2, MIB SMUX, el recurso de *host* MIB y la MIB UCD-SNMP. Estas MIB están localizadas de manera predeterminada en el directorio `/usr/local/share/snmp/mibs`. Sin embargo, el agente debe funcionar aún correctamente, si es incapaz de ubicar estos archivos.

Desde la perspectiva de la administración de red, el agente puede hacer seguimiento a los parámetros críticos del sistema operativo (como lo es el promedio de carga del sistema) y proporcionar

alertas cuando estos recursos alcanzan un umbral crítico. El agente puede hacer lo siguiente:

- seguimiento al uso del espacio del disco
- seguimiento a los procesos del sistema
- seguimiento a la carga del sistema
- invocar comandos UNIX y secuencias de comandos shell
- seguimiento de la información y estado del agente
- proporcionar acceso a objetos MIB clave

La siguiente sección explora cada una de estas características en mayor detalle, incluido cómo configurar al agente para hacerles seguimiento a estos recursos y cómo abordar proactivamente los problemas de limitación de recursos del sistema.

Archivo de configuración de agente

El agente requiere un archivo de configuración para operar. Este archivo está localizado de manera predeterminada en el directorio `usr/local/share/snmp`. Éste contiene una o más directivas de acción, y cada línea tiene una única palabra clave con argumentos asociados. La palabra clave puede clasificarse en una de las categorías siguientes: *seguimiento*, *seguridad*, *extensible* e *informativa*, como se muestra en la tabla 7.2. Las directivas de seguimiento (**proc**, **disk** y **load**) incluyen las utilizadas para seguir recursos específicos, como el espacio del disco o la carga del sistema. Las palabras clave de seguridad (**com2sec**, **group**, **view** y **acces**) se utilizan para definir varias capas de seguridad para tener acceso al agente, que pueden incluir listas de control de acceso (ACL) y varias vistas de los objetos MIB. Las palabras clave extensibles (**exec** y **pass**) definen formas de agregar funcionalidad al agente, sin cambiar el software del agente. Finalmente, las palabras clave informativas (**syslocation** y **syscontact**) asignan información estática a los objetos MIB. Cada una de las palabras clave tiene un número fijo de parámetros requeridos y puede tener también argumentos opcionales adicionales.

Tabla 7.2 Comandos del agente UCD ordenados por categoría.

Categoría	Palabra clave	Descripción
Seguimiento	proc	Verifica si los procesos especificados están corriendo
	disk	Determina el espacio de disco disponible en el sistema de archivo especificado
	load	Hace un seguimiento al promedio de carga del sistema
Seguridad	com2sec	Especifica una transformación entre nombre de la fuente/comunidad y el nombre de seguridad
	group	Define una transformación entre un nombre modelo/ seguridad y un grupo
	view	Define una o más vistas específicas en las MIB del agente
	access	Parámetro de configuración de seguridad
Extensible	exec	Ejecuta el comando específico y pone a disposición la salida a través de la MIB
	pass	Pasa un árbol MIB al programa especificado que pueda manipular los objetos MIB
Informativas	syslocation	Representa la localización física del dispositivo del agente
	syscontact	Representa la persona de contacto para el dispositivo del agente

Para insertar comentarios, se coloca el carácter # al comienzo de la línea.

Cuando el agente arranca por primera vez, lee el archivo de configuración. Si se han realizado cambios al archivo después de que el agente ha arrancado, el agente puede ser instruido para que relea el archivo de configuración de dos maneras. En la primera, el objeto MIB `versionUpdateConfig` puede colocarse en `1`. Por ejemplo, empleando la herramienta UCD `snmpset`, el siguiente comando forzaría al agente en el sistema `rodin` a procesar otra vez el archivo de configuración:

```
snmpset rodin private .1.3.6.1.4.1.ucdavis.version.version-
UpdateConfig.0 i 1
```

En la segunda, la señal `HUP` puede enviarse al proceso del agente mediante el uso del comando **kill**. Antes de emitir este comando, debemos obtener la identificación del proceso del agente. Esto puede hacerse, por ejemplo, mediante la emisión del comando **ps**. Suponiendo que la identificación del proceso que se encontró es `304`, el siguiente comando sería funcionalmente equivalente al comando **snmpget:**

```
# kill -HUP 304
```

Algunas de las palabras clave de los agentes, cuando se ejecutan, hacen un seguimiento a los recursos que modifican ciertos objetos MIB que indican el estado de retorno u otra información adicional. Estos objetos son parte de la MIB del agente y se accede a ellos desde la rama privada UCD (`2021`). La ruta de inicio de la MIB es:

```
iso.org.dod.internet.mgmt.enterprises.ucdavis
```

o

```
1.3.6.1.4.1.2021
```

Además, si se encuentra que el agente debe ser reiniciado debido a algún otro problema operacional, como una caída, entonces se invoca la secuencia de comandos estándar de arranque: `snmpd`. Por ejemplo, para reiniciar el agente se emitiría el siguiente comando:

```
# /etc/rc.d/init.d/snmpd restart
Shutting down snmpd:
[ OK ]
Starting snmpd:
[ OK ]
```

Si el agente arrancó exitosamente, se mostrara la cadena `[ OK ]`; de otra manera, se mostrará una cadena de error.

Nota

La secuencia de comandos `snmpd` soporta comandos adicionales. Éstos incluyen **stop**, **start** y **status**. La opcion **stop** terminará al agente si está corriendo, mientras que la opcion **start** iniciará al agente si todavía no está activo. La opción final, **status**, verifica que el agente esté corriendo.

Seguimiento de la carga del sistema

El agente UCD proporciona la capacidad de hacer un seguimiento al promedio de carga del sistema a través del comando **load** del agente. El promedio de carga del sistema es el número promedio de trabajos (o programas) en la cola de ejecución. El agente suministra la misma información básica que los comandos **uptime** y **w** de UNIX. La sintaxis de este comando incluye:

```
load [1 minute max] [5 minute max] [15 minute max]
```

El comando **load**, seguido por `1` máximo, `5` máximo o `15` máximo, instruirá al agente para hacer un seguimiento a la carga del sistema para estas cantidades de tiempo. Si la carga del sistema excede el máximo para los promedios de ejecución de un minuto, cinco minutos o quince minutos, entonces el agente pondrá el objeto `loadaveErrorFlag` en `1` e incluirá un mensaje descriptivo de error en el objeto `loadaveErrMessage`. Si la cadena `maximum load` (carga máxima) no se especifica, entonces se usa el valor predefinido de `1` minuto.

El comando **load** es una manera excelente de hacer un seguimiento a la utilización del sistema a través de una gran cantidad de sistemas. Establecer el umbral de carga del sistema puede ser un poco delicado porque ciertos sistemas pueden necesitar umbrales diferentes. Por ejemplo, una estación de trabajo puede experimentar cargas altas en el sistema porque el usuario acabó de iniciar una aplicación intensiva de CPU. De otro lado, un usuario que inicia la misma aplicación en un servidor con una gran cantidad de usuarios podría plantear un problema significativo. Además, algunos sistemas pueden tener siempre una carga más alta que otros sistemas pero, debido a que se espera que el sistema se use de esta manera, no representa un problema.

Para sistemas críticos, un promedio de carga mayor que `2` o `3` puede indicar un problema severo que debe abordarse de una

manera oportuna. En el ejemplo siguiente, el agente se ha configurado para hacer un seguimiento del promedio de carga de un minuto, con un umbral de 2:

```
load 2
```

Si se alcanza este límite, la MIB del agente se actualizará apropiadamente:

```
enterprises.ucdavis.loadTable.laEntry.laErrorFlag.1 = 1
enterprises.ucdavis.loadTable.laEntry.laErrorFlag.2 = 1
enterprises.ucdavis.loadTable.laEntry.laErrorFlag.3 = 0
enterprises.ucdavis.loadTable.laEntry.laErrMessage.1 = "1
  min Load Average too high (= 2.50)"
enterprises.ucdavis.loadTable.laEntry.laErrMessage.2 = "5
  min Load Average too high (= 2.26)"
enterprises.ucdavis.loadTable.laEntry.laErrMessage.3 = ""
```

En este caso, la MIB muestra que los dos promedios de carga, el de un minuto y el de cinco, excedieron el umbral límite de 2. La MIB muestra tres instancias de `laErrorFlag` y de `laErrMessage` porque el promedio de carga se mide en términos de promedios de uno, cinco y 15 minutos. Por tanto, se crean tres instancias separadas de objetos MIB.

El agente también proporciona acceso tanto a los promedios actuales de carga de sistema como a los parámetros de configuración del comando **load**. Por ejemplo, lo siguiente se obtuvo del agente cuando la carga estaba por encima de la marca 2:

```
enterprises.ucdavis.loadTable.laEntry.laNames.1 = "Load-1"
enterprises.ucdavis.loadTable.laEntry.laNames.2 = "Load 5"
enterprises.ucdavis.loadTable.laEntry.laNames.3 = "Load-15"
enterprises.ucdavis.loadTable.laEntry.laLoad.1 = "2.50" Hex: 32 2E 35 30
enterprises.ucdavis.loadTable.laEntry.laLoad.2 = "2.26" Hex: 32 2E 32 36
enterprises.ucdavis.loadTable.laEntry.laLoad.3 = "1.38" Hex: 31 2E 33 38
enterprises.ucdavis.loadTable.laEntry.laConfig.1 = "2.00" Hex: 32 2E 30 30
enterprises.ucdavis.loadTable.laEntry.laConfig.2 = "2.00" Hex: 32 2E 30 30
enterprises.ucdavis.loadTable.laEntry.laConfig.3 = "2.00" Hex: 32 2E 30 30
```

Como se puede observar, los promedios de la carga para uno y cinco minutos están por encima de 2. Además, el objeto `laConfig` está establecido en `2.00` para cada nivel del umbral promedio de carga. El objeto `laNames` es simplemente una etiqueta que se usa para describir cada ventana de promedio de carga. Esta información se obtuvo directamente del archivo de configuración del agente.

Seguimiento del espacio del disco

El agente puede realizar un seguimiento al consumo del espacio del disco de cualquiera de los sistemas de archivos montados con el comando de la palabra clave **disk**, usando la siguiente sintaxis:

```
disk sistema-de-archivo [mínimo]
```

El agente hará un seguimiento al sistema de archivos especificado y determinará si el espacio total del disco disponible es menor que el mínimo. Si es así, el objeto MIB `diskErrorFlag` será colocado en `1`, y el objeto MIB `diskErrorMsg` contendrá un mensaje descriptivo. Si no se proporciona un mínimo, el agente usa el valor predeterminado de `100K`. El valor mínimo se almacena en el objeto `dskMinimum` y se coloca o bien en el mínimo especificado desde el archivo de configuración o en el predeterminado, si no se encuentra ningún valor mínimo en el archivo de configuración.

Por ejemplo, considérense los siguientes comandos del agente:

```
disk /     100000
disk /usr 1000
```

El primer comando **disk** hará un seguimiento del sistema de archivo `/(root)` y comparará el valor mínimo de 100 megabytes (M). Si no se suministra ningún argumento de tamaño, se utilizará un valor de 100 kilobytes (el predeterminado). Este valor se compara con el espacio total del disco que está disponible. El valor predeterminado puede ser sobrepuesto. El segundo comando realizará un seguimiento al sistema de archivos `/usr` y comparará el espacio disponible con el valor mínimo de 1,000 Kb. Debido a que el valor mínimo está en unidades Kb, se debe agregar tres ceros al final para tener el valor en bytes. En este caso, se tiene 1,000,000 de bytes. Cuando se consulta al agente, retorna lo siguiente:

```
enterprises.ucdavis.diskTable.dskEntry.dskMinimum.1 =
  100000
enterprises.ucdavis.diskTable.dskEntry.dskMinimum.2 = 1000
enterprises.ucdavis.diskTable.dskEntry.dskAvail.1 = 56924
enterprises.ucdavis.diskTable.dskEntry.dskAvail.2 = 32545
enterprises.ucdavis.diskTable.dskEntry.dskErrorFlag.1 = 1
enterprises.ucdavis.diskTable.dskEntry.dskErrorFlag.2 = 0
enterprises.ucdavis.diskTable.dskEntry.dskErrorMsg.1 = "/:
  less than 100000 free (= 56924)"
enterprises.ucdavis.diskTable.dskEntry.dskErrorMsg.2 = ""
```

Se notará que están definidos dos conjuntos de objetos MIB. El primer conjunto corresponde al sistema de archivos `root (/)`, porque éste aparece primero en el archivo de configuración del agente. El segundo conjunto es para el sistema de archivos `/usr`, porque viene después de la primera entrada dentro del archivo de configuración.

Como se puede observar, los objetos `dskMinimum` están establecidos como corresponde; la instancia `dskMinimum.1`, se refiere al sistema de archivos /, y está establecido en 100Mb (100,000,000 bytes), mientras que el segundo, `diskMinimum.2`, el cual representa al sistema de archivos `/usr`, se establece en 1 Mb (1,000,000 bytes). El agente también almacena el espacio disponible del disco de cada sistema de archivo al que le hace seguimiento en el objeto `dskAvail`, uno para cada sistema de archivo. En este caso, el sistema de archivos / es aproximadamente de 56Mb, mientras que el sistema de archivos `/usr` es de aproximadamente 32Mb. El objeto `dskErrorFlag.1` para el sistema de archivos se ha establecido porque el sistema de archivo raíz es de menos de 100Mb y `dskErrorMsg.1` contiene más información descriptiva. En este caso, el objeto `dskErrorMsg.2` no contiene ninguna información. El objeto `dskErrorFlag.2` no está colocado porque el sistema de archivos `/usr` contiene suficiente espacio para satisfacer la prueba del espacio de disco.

La MIB también contiene otra información que puede usarse cuando se hace un seguimiento al espacio del disco. La información se compara con la salida del comando UNIX **df** (disco libre). Por ejemplo, el comando **df –k** incluye un resumen del espacio del disco del sistema de archivo montado. La salida que se presenta a continuación incluye una lista parcial de los sistemas de archivo reportados:

```
Filesystem    1k blocks      Used Available Use%
Mounted on
/dev/hdb1         82303     25286     56935  31%
/dev/hdb2        232141    199364     32545  86%
/usr
```

Cuando se consulta al agente, se encuentra lo siguiente:

```
enterprises.ucdavis.diskTable.dskEntry.dskPath.1 = "/" Hex: 2F
enterprises.ucdavis.diskTable.dskEntry.dskPath.2 = "/usr" Hex: 2F 75 73 72
enterprises.ucdavis.diskTable.dskEntry.dskDevice.1 = "/dev/hdb1"
enterprises.ucdavis.diskTable.dskEntry.dskDevice.2 = "/dev/hdb2"
```

(*continúa*)

```
enterprises.ucdavis.diskTable.dskEntry.dskTotal.1 = 82303
enterprises.ucdavis.diskTable.dskEntry.dskTotal.2 = 232141
enterprises.ucdavis.diskTable.dskEntry.dskAvail.1 = 56935
enterprises.ucdavis.diskTable.dskEntry.dskAvail.2 = 32545
enterprises.ucdavis.diskTable.dskEntry.dskUsed.1 = 25286
enterprises.ucdavis.diskTable.dskEntry.dskUsed.2 = 199364
enterprises.ucdavis.diskTable.dskEntry.dskPercent.1 = 31
enterprises.ucdavis.diskTable.dskEntry.dskPercent.2 = 86
```

Nótese que la MIB contiene toda la información que proporciona el comando **df**. Ésta incluye los puntos de montaje, los sistemas de archivos, el espacio total del disco, el espacio disponible del disco, el espacio usado del disco y el porcentaje del espacio usado del disco. Esta información puede ser extremadamente útil porque, por ejemplo, es posible hacerle un seguimiento al uso del espacio del disco en porcentajes en lugar de bytes, empleando los objetos `dskPercent`. Además, los objetos restantes pueden usarse como un mecanismo automatizado para inventariar las capacidades de almacenamiento del sistema. La sección "MIB Linux" incluye más información acerca de éstos y otros objetos MIB del agente.

Seguimiento a los procesos del sistema

El comando **proc** puede hacer un seguimiento a servicios específicos de aplicación o a procesos estándar UNIX. El comando utiliza la siguiente sintaxis:

```
proc process [maximum limit] [minimum limit]
```

Este comando es seguido por un nombre válido de proceso o un comando que debe ser seguido por el agente. Está directiva se utiliza para hacer un seguimiento a los procesos críticos, o a un conjunto de procesos, que deben estar corriendo continuamente en el sistema. Si el agente no encuentra al proceso corriendo en el sistema local, se coloca una bandera de error en el objeto MIB `prvTable.prEntry.prErrorFlag`, y se coloca un mensaje descriptivo en el objeto MIB `procTable.prEntry.prErrMessage`. El agente utiliza el comando UNIX **ps -e** para buscar procesos en ejecución, como se especifica con el comando **proc**. Por ejemplo, el siguiente comando puede usarse para asegurar que el proceso `mountd` está activo:

```
proc mountd
```

Adicionalmente, se puede incluir un valor *máximo* y uno *mínimo* para refinar aún más la cantidad de procesos que deberían estar corriendo. El máximo indica cuántos procesos deben estar corriendo de manera concurrente, mientras el mínimo determina el número más pequeño que debe estar corriendo. Por ejemplo, puesto que se acostumbra tener procesos múltiples `mountd` en un sistema, el agente puede configurarse para hacer un seguimiento del número exacto. Como hipótesis, supongamos que se habilita el sistema para soportar cuatro instancias de `mountd`:

```
proc mountd 4 4
```

Si el agente detecta que menos de cuatro o mucho más de cuatro procesos `mountd` están corriendo en este momento, actualiza apropiadamente la MIB.

Si los valores máximo y mínimo no se especifican, se supone que el número predeterminado es cero. En la práctica, el uso tanto de la opción máximo como de la opción mínimo proporciona una forma flexible de abordar los requerimientos específicos de seguimiento.

El comando

```
proc ntalkd 4
```

le ordena al agente que haga el seguimiento del proceso `ntalkd` y actualice la MIB si más de cuatro están corriendo en el momento. Por consiguiente, la opción máximo suministra una forma conveniente de asegurar que sólo una cierta cantidad de procesos tienen permitido correr actualmente. Si se quiere ir un paso adelante, se automatiza al agente para tomar una acción (por ejemplo, matar el proceso), cuando se encuentran los procesos `ntalkd`. Como ningún mínimo fue listado, el agente no actuará cuando no encuentre corriendo procesos `ntalkd`.

La selección de los procesos UNIX a los que debe hacerse seguimiento con la palabra clave `proc` es sobre todo un problema de determinar cuáles servicios son críticos para la operación de su entorno. Si los sistemas de computación soportan software editorial, pueden ser de mayor interés *spoolers* de impresión y las aplicaciones de procesamiento de palabras, mientras los sistemas que soportan software para entrar órdenes en línea pueden necesitar hacer seguimiento a los procesos del servidor de base de datos.

Por otro lado, se puede plantear que todo lo que corre en un sistema es crítico y debe hacérsele seguimiento. A primera vista, esto parece ser una estrategia razonable. Desafortunadamente, también puede llevar a otros problemas y ser exagerado en la mayoría de las situaciones. Si se quiere hacer un seguimiento a todos los procesos del sistema, ¿se debe también hacer un seguimiento a los procesos temporales, como los servicios del trabajo en red? Si es así, ¿a cuáles? Además, mantener el archivo de configuración del agente de estos procesos puede ser una tarea rutinaria, en el mejor de los casos, porque debe ser actualizado cuando se agregan o remueven nuevas aplicaciones y software del sistema. El rendimiento del agente también se verá afectado. Hacer seguimiento a un gran número de procesos puede dar como resultado una saturación del agente, la cual debería evitarse.

Supóngase que se ha configurado al agente para hacer un seguimiento del proceso `mountd` y de cuatro procesos `ntalkd`. El siguiente ejemplo muestra los valores de la MIB cuando el agente detecta que ninguno de los procesos `mountd` o `ntalkd` se están ejecutando:

```
enterprises.ucdavis.procTable.prEntry.prErrorFlag.1 = 1
enterprises.ucdavis.procTable.prEntry.prErrorFlag.2 = 1
enterprises.ucdavis.procTable.prEntry.prErrMessage.1 = "No
  mountd process running."
enterprises.ucdavis.procTable.prEntry.prErrMessage.2 = "Too
  few ntalkd running (# = 0)"
```

Seguridad del agente

El agente UCD soporta los métodos de seguridad SNMPv1, SNMPv2 y SNMPv3, y se configura usando las palabras clave de seguridad listadas en la tabla 7.2. El modelo de seguridad usual implementado por el agente incluye el modelo de control de acceso basado en la vista (VACM); sin embargo, los modelos de seguridad antiguos de SNMPv1 y SNMPv2 también son soportados. Este modelo VACM fue introducido con las especificaciones del SNMPv3 y, en pocas palabras, proporciona seguridad basada en diferentes vistas y niveles de usuarios que controlan el acceso a objetos individuales, grupos o a una MIB completa.

La palabra clave **com2sec** especifica una transformación entre un nombre de fuente/comunidad y un nombre de seguridad, y tiene la siguiente sintaxis:

```
com2sec security_name source community_string
```

donde `security_name` puede ser cualquier cadena y se usa en definiciones subsecuentes de políticas de seguridad. El parámetro `source` puede ser un nombre de *host*, una dirección IP, una red o la palabra clave reservada **default**. La palabra **default** es un comodín y corresponderá a cualquier fuente. Cuando se especifica la red, se soportan dos formatos diferentes: la dirección con la máscara de dirección y la dirección de red con un número activo de bits. Por tanto,

```
134.111.82.0/255.255.255.0
```

y

```
134.111.82.0/24
```

son equivalentes y pueden usarse de manera intercambiable. `community_string` representa la contraseña de comunidad que será suministrada por la entidad administradora. El siguiente es un ejemplo de la entrada **com2sec**:

```
com2sec public-sec-name default public
```

Esta entrada define un nombre de seguridad de `public-sec name`, y cualquier fuente que use la cadena de comunidad `public` tiene acceso permitido. El siguiente ejemplo proporciona control adicional:

```
com2sec priv-sec-name 134.111.80.0/24 private-80
```

En este caso, se define una nueva cadena de seguridad de `priv sec-name`, la cual limita el acceso a cualquier dispositivo con una dirección fuente de `134.111.80.0` y que tenga `private-80` como contraseña de comunidad.

La directiva de palabra clave **group** define una transformación entre un nombre de modelo/seguridad y un grupo:

```
group group_name available_model security_name
```

`group_name` es una cadena que puede representar una colección de usuarios o cualquier otra definición de grupo. `available model` es uno de los modelos de seguridad definidos soportados por el agente. Éstos incluyen **any**, **v1**, **v2c** y **usm.** La palabra clave **any** es un comodín que indica cualquiera de los otros modelos. El modelo **v1** está asociado con la cadena de comunidad `SNMPv1`, donde un único nombre de cadena se usa como la señal (*token*) de autenticación. El modelo **v2c** emplea el

uso de la autenticación de comunidad, basado en llaves de seguridad públicas y/o privadas. El modelo **usm**, conocido como el Modelo de Seguridad del Usuario (USM), fue introducido con el SNMPv3 y contiene especificaciones que exigen que los usuarios sean definidos en un agente SNMP para propósitos de autenticación.

El propósito principal de la palabra clave **group** es definir aún más niveles de control sobre grupos individuales de usuarios u otras entidades. Por ejemplo, considérese lo siguiente:

```
group anybody any public-sec name
```

La entrada en este ejemplo define un grupo llamado `anybody` que indica básicamente que se permite cualquier acceso que utilice todos los modelos de seguridad, siempre que se utilice la cadena de comunidad asociada con el `public-sec name`. Por tanto, en el ejemplo

```
group admin1 usm priv-sec-name
```

un grupo llamado `admin1` se define con instrucciones estrictas sobre el acceso, porque éste usa un modelo **usm** con el nombre de seguridad `priv sec-name`. En este punto, la sintaxis y la semántica exactas de los modelos de seguridad **v2c** y **usm** son indefinidas. Sin embargo, para el propósito de este análisis, se supone que lo que sea que se necesite, el software asociado del administrador de red es responsable de proporcionar la información apropiada o correcta al agente.

La palabra clave **view** define una o más vistas específicas de las MIB del agente. Ésta es una característica muy poderosa porque permite a la función de administración de la red y a los dispositivos relacionados de red ser particionados en formas interesantes y beneficiosas. Por ejemplo, un ISP puede proporcionar acceso a sus clientes, a sus dispositivos internos de trabajo en red o a interfaces específicas. Suministrando acceso sólo a esas interfaces y redes para las cuales el cliente necesita acceso de administración, la seguridad de red y administración es preservada para el ISP, pero los clientes pueden llevar a cabo soluciones de problemas avanzados y hacer seguimiento de rendimiento. Todo esto se puede conseguir usando diferentes vistas MIB del agente contenidas en los dispositivos de trabajo en red.

La sintaxis del comando **view** es la siguiente:

```
view name type mib_tree [mask]
```

El siguiente argumento del comando **view** es el nombre de la vista especificada. El nombre puede ser cualquier valor de cadena, pero debería ser una descripción de los objetos MIB o de grupos definidos en la vista. **name** es seguido por el tipo de vista: **included** o **excluded**. La palabra clave **included** significa que el objeto `mib_tree` se incluye en la vista, mientras que **excluded** significa que el objeto `mib_tree` no está incluido en la vista. `mib tree` representa la definición de la MIB en el agente para el cual la vista será de acceso permitido o denegado. El objeto `mib_tree` puede incluir un objeto MIB específico o un grupo. Por ejemplo, `.1`, `system` y `.iso.org.dod.internet.mgmt.mib-2.interfaces` son todas definiciones válidas del objeto MIB. El siguiente es un ejemplo de una vista válida:

```
view system_group include system fe
```

Este ejemplo define una nueva vista llamada `system_group`, la cual permite el acceso a todo el grupo `system` MIB-II. El argumento final es la opción `mask`, la cual controla más precisamente los objetos que se aplican a la vista. El valor `mask` enmascara un conjunto particular de cadenas identificadoras de objetos. La máscara proporciona una forma simple de habilitar o inhabilitar partes (o todo) el árbol MIB. Cada bit de la máscara se compara con el correspondiente identificador de objeto MIB. Si el bit en la máscara es `1`, entonces el objeto se usa. Si el bit es `0`, entonces cualquier valor puede aparecer en la posición del identificador del objeto. El valor `0` se usa como comodín. Dado el ejemplo precedente, se examina por qué la máscara `fe` permitirá el acceso al grupo `system`. Primero, debemos considerar que la ruta OID del grupo `system` puede expresarse en notación puntuada, como se muestra:

```
system = 1.3.6.1.4.1.1
```

A continuación, necesitamos convertir la máscara `fe` a binario:

```
fe (hex) = 1 1 1 1 1 1 1 0 (binary)
```

Ahora necesitamos comparar la cadena OID con la máscara:

```
1 3 6 1 4 1 1     (OID value)
1 1 1 1 1 1 1 0   (Mask value)
```

Comparando estos valores, se puede ver que todos los objetos arriba del grupo *system*, están enmascarados. Por tanto, el acceso está permitido solamente desde abajo del grupo *system*.

Extensión del agente

Una de las características más poderosas del agente UCD es que se puede extender para incluir la ejecución de comandos externos. Estos comandos pueden ser secuencias de comandos shell o programas binarios, los cuales pueden especificarse en la configuración del agente. La salida, a través de la salida estándar, es capturada en la MIB del agente, de modo que la información puede estar disponible a través de las operaciones SNMP *get-requests* y *getnext-requests*.

Para invocar comandos externos UNIX o secuencias de comandos shell desde el agente, se usa el comando **exec**, el cual tiene la siguiente sintaxis:

```
exec label program [args]
exec MIB program [args]
```

En la primera forma, el comando **exec** se utiliza con una etiqueta, un programa y unos argumentos opcionales. La etiqueta se usa como un identificador, el cual es útil cuando se explora la MIB del agente. El programa es el binario real UNIX o secuencia de comandos shell que el agente invocará, con los argumentos especificados en la línea de comandos listados en el campo `args`. Cuando el agente ejecuta este comando, la salida es colocada en la MIB del agente bajo la tabla `extEntry`. En el siguiente ejemplo, el comando **exec** ejecuta el comando **uptime** para determinar el nivel de ejecución del sistema local:

```
exec uptime /usr/bin/uptime
```

El comando **uptime** muestra por cuánto tiempo el sistema ha estado corriendo desde que se inició por última vez, y el promedio de carga del sistema. La cadena **uptime** es la etiqueta, y cuando el agente invoca el comando **uptime**, la salida es colocada en la MIB y está disponible para ser vista. Usando la herramienta `snmpwalk`, descrita en el capítulo 8, "Herramientas SNMP", se obtuvo del agente la siguiente salida de ejemplo:

```
enterprises.ucdavis.extTable.extEntry.extIndex.1 = 1
enterprises.ucdavis.extTable.extEntry.extNames.1 = "uptime"
enterprises.ucdavis.extTable.extEntry.extCommand.1 =
   "/usr/bin/uptime"
enterprises.ucdavis.extTable.extEntry.extResult.1 = 0
enterprises.ucdavis.extTable.extEntry.extOutput.1 = 10:47pm
   up 2:09, 3 users, load average: 0.57, 0.37, 0.22."
enterprises.ucdavis.extTable.extEntry.extErrFix.1   0
```

Nótese que el objeto `extResult` contiene un `0`. Esto se debe a que el comando se completó exitosamente. Además, tanto la etiqueta `uptime` como el comando **/usr/bin/uptime** se almacenan en los objetivos `extNames` y `extCommand`, respectivamente.

Una característica de esta versión de **exec** es que sólo una única línea de salida se guarda para la MIB. Como resultado, cualquier salida después de la primera línea será descartada por el agente. Esto limita, un poco, la utilidad de este comando. Afortunadamente, la segunda forma del comando **exec** puede usarse para capturar la salida cuando hay más de una línea.

Con la segunda forma, el comando **exec** incluye una definición de la MIB que se usará para almacenar la información del comando **exec** y la salida completa del comando que fue ejecutado por el agente. Por ejemplo, considérese el siguiente comando:

```
exec .1.3.6.1.4.1.2021.51 ps /bin/ps -e
```

Este comando **exec** ejecutará el comando **ps** y asignará la salida a la rama MIB.

```
.iso.org.dod.internet.private.ucdavis.51
```

El comando **ps** muestra información del proceso sistema/aplicación. La opción `-e` se utiliza para recolectar información respecto a todos los procesos que corren en el sistema.

Para examinar la salida de **ps**, simplemente hacemos un recorrido (usando `snmpwalk`, por ejemplo) en la rama MIB `2021.51`. Nótese que para propósitos de lectura, la cadena OID antes que la cadena `enterprises` se omitió de la salida que sigue.

Por tanto, el comando

```
snmpwalk socrates public .iso.org.dod.internet.private.enterprises.ucdavis.51
```

dará los siguientes resultados:

```
enterprises.ucdavis.51.1.1 = 1
enterprises.ucdavis.51.2.1 = "ps" Hex: 70 73
enterprises.ucdavis.51.3.1 = "/bin/ps -e"
enterprises.ucdavis.51.100.1 = 0
enterprises.ucdavis.51.101.1 = "  PID TTY      TIME CMD."
enterprises.ucdavis.51.101.2 = "    1 ?    00:00:03 init."
enterprises.ucdavis.51.101.3 = "    2 ?    00:00:00 kflushd."
enterprises.ucdavis.51.101.4 = "    3 ?    00:00:00 kpiod."
enterprises.ucdavis.51.101.5 = "    4 ?    00:00:00 kswapd."
enterprises.ucdavis.51.101.6 = "    5 ?    00:00:00 mdrecoveryd."
enterprises.ucdavis.51.101.7 = "  107 ?    00:00:00 apmd."
```

(*continúa*)

```
enterprises.ucdavis.51.101.8 = " 257 ?      00:00:00 portmap."
enterprises.ucdavis.51.101.9 = " 307 ?      00:00:02 syslogd."
enterprises.ucdavis.51.101.10 = " 318 ?     00:00:00 klogd."
enterprises.ucdavis.51.101.11 = " 332 ?     00:00:00 atd."
enterprises.ucdavis.51.101.12 = " 346 ?     00:00:00 crond."
enterprises.ucdavis.51.101.13 = " 364 ?     00:00:00 inetd."
enterprises.ucdavis.51.101.14 = " 394 ?     00:00:00 arpwatch."
enterprises.ucdavis.51.101.15 = " 408 ?     00:00:00 named."
enterprises.ucdavis.51.101.16 = " 422 ?     00:00:00 lpd."
enterprises.ucdavis.51.101.17 = " 440 ?     00:00:00 rpc.statd."
enterprises.ucdavis.51.101.18 = " 451 ?     00:00:00 rpc.rquotad."
enterprises.ucdavis.51.101.19 = " 462 ?     00:00:00 rpc.mountd."
enterprises.ucdavis.51.101.20 = " 477 ?     00:00:00 nfsd."
enterprises.ucdavis.51.101.21 = " 478 ?     00:00:00 nfsd."
enterprises.ucdavis.51.101.22 = " 479 ?     00:00:00 nfsd."
enterprises.ucdavis.51.101.23 = " 480 ?     00:00:00 nfsd."
enterprises.ucdavis.51.101.24 = " 481 ?     00:00:00 nfsd."
enterprises.ucdavis.51.101.25 = " 485 ?     00:00:00 lockd."
enterprises.ucdavis.51.101.26 = " 486 ?     00:00:00 rpciod."
enterprises.ucdavis.51.101.27 = " 503 ?     00:00:00 amd."
enterprises.ucdavis.51.101.28 = " 540 ?     00:00:00 sendmail."
enterprises.ucdavis.51.101.29   " 555 ?     00:00:00 gpm."
enterprises.ucdavis.51.101.30   " 599 ?     00:00:01 xfs."
enterprises.ucdavis.51.101.31 = " 651 ?     00:00:00 innd."
enterprises.ucdavis.51.101.32   " 654 ?     00:00:00 actived."
enterprises.ucdavis.51.101.33 = " 695 tty1  00:00:00 mingetty."
enterprises.ucdavis.51.101.34   " 696 tty2  00:00:00 mingetty."
enterprises.ucdavis.51.101.35 = " 697 tty3  00:00:00 mingetty."
enterprises.ucdavis.51.101.36   " 698 tty4  00:00:00 mingetty."
enterprises.ucdavis.51.101.37 = " 699 tty5  00:00:00 mingetty."
enterprises.ucdavis.51.101.38 = " 700 tty6  00:00:00 mingetty."
enterprises.ucdavis.51.101.39   " 701 ?     00:00:00 prefdm."
enterprises.ucdavis.51.101.40 = " 703 ?     00:00:00 update."
enterprises.ucdavis.51.101.41 = " 705 ?     00:05:07 X."
enterprises.ucdavis.51.101.42 = " 708 ?     00:00:00 prefdm."
enterprises.ucdavis.51.101.43 = " 713 ?     00:00:00 Default."
enterprises.ucdavis.51.101.44   = " 726 ?   00:00:00 gnome-session."
enterprises.ucdavis.51.101.45 = " 758 ?     00:01:55
  enlightenment."
enterprises.ucdavis.51.101.46   = " 773 ?   00:00:01 xscreensaver."
enterprises.ucdavis.51.101.47 = " 775 ?     00:00:01 panel."
enterprises.ucdavis.51.101.48 = " 777 ?     00:00:00 gnome-name-
  serv."
enterprises.ucdavis.51.101.49 = " 781 ?     00:00:00
  gen_util_applet."
enterprises.ucdavis.51.101.50 = " 784 ?     00:01:09
  gnomepager_appl."
enterprises.ucdavis.51.101.51 = " 789 ?     00:00:05 gnome-
  terminal."
```

(continúa)

```
enterprises.ucdavis.51.101.52 = " 791 ?      00:00:07 gnome-
  terminal."
enterprises.ucdavis.51.101.53 = " 793 ?      00:00:02 gmc."
enterprises.ucdavis.51.101.54   " 795 ?      00:00:01 gtcd."
enterprises.ucdavis.51.101.55   = " 796 ?      00:00:00 gnome-pty-
  helpe."
enterprises.ucdavis.51.101.56   = " 797 ?      00:00:00 gnome pty-
  helpe."
enterprises.ucdavis.51.101.57 = " 798 pts/0 00:00:00 bash."
enterprises.ucdavis.51.101.58   " 799 pts/1 00:00:00 bash."
enterprises.ucdavis.51.101.59 = " 949 ?      00:00:00 in.telnetd."
enterprises.ucdavis.51.101.60   " 950 pts/2 00:00:00 login."
enterprises.ucdavis.51.101.61 = " 951 pts/2 00:00:00 bash."
enterprises.ucdavis.51.101.62 = " 1569 pts/0 00:00:01 snmpd."
enterprises.ucdavis.51.101.63     " 1605 pts/2 00:00:00 snmpwalk."
enterprises.ucdavis.51.101.64 = " 1606 pts/0 00:00:00 ps."
enterprises.ucdavis.51.102.1    0
```

Podemos utilizar algunos de los objetos MIB hallados en el grupo MIB `ucdavis.51` para ayudar a identificar la salida del comando **ps**. Los objetos `ucdavis.51.2.1` y `ucdavis.51.3.1` contienen la etiqueta de comando y el comando en sí, como se especifica en el archivo de configuración del agente. Los objetos MIB restantes (`ucdavis.51.101.1` a `ucdavis.51.101.23`) representan la salida del comando. El objeto MIB `ucdavis.51.101.1` también muestra el estado de retorno del comando **ps**. En este ejemplo, un `0` indica que el comando **ps** se ejecutó exitosamente. Como se puede observar, el comando **exec** puede extender la cobertura de seguimiento del agente para incluir herramientas adicionales del sistema operativo y funciones de aplicación. Ésta es tal vez una de las características más poderosas del agente Linux.

Uso de la facilidad `pass`

Otra característica poderosa del agente UCD es el comando **pass**. Este comando puede dar el control de un árbol MIB a un programa que el agente ejecuta. Esto da al agente la capacidad de alterar dinámicamente objetos MIB, sin requerir un desarrollo personalizado del agente. La figura 7.2 muestra un flujo de alto nivel. En general, el agente recibe una o más solicitudes SNMP que tienen acceso a parte del árbol MIB asociado con el comando **pass**. Como resultado, el agente ejecuta los últimos dos argumentos del comando **pass**, en este caso `/bin/sh` y `/usr/local/passtest`, respectivamente. Cuando esta secuencia de comandos se ejecuta, el agente suministra parámetros adicionales a la secuencia de comandos `passtest` que contiene la OID a la que se

busca acceso, el tipo, y el valor asociado. El agente captura la salida de la secuencia de comandos y la coloca en la MIB.

Para utilizar la opción `pass`, se crea un programa simple que muestra la información requerida del objeto MIB y realiza cualquier trabajo que se necesite. Por ejemplo, considérese este comando:

```
pass .1.3.6.1.4.1.2021.255 /bin/sh /usr/local/passtest
```

La opción `pass` incluye un árbol de objeto MIB y el comando que se ejecutará cuando se acceda a la MIB. En este caso, el objeto MIB `1.3.6.1.4.1.2021.255` se pasa realmente a la secuencia de comandos `passtest`.

Más específicamente, el agente llamará a la secuencia de comandos `passtest` en una de tres formas, dependiendo del modo de operación solicitado.

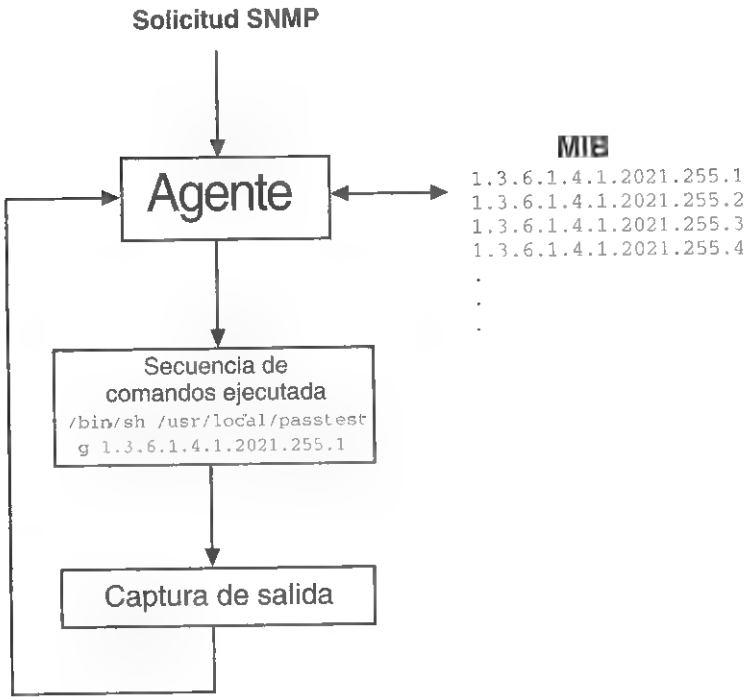


Figura 7.2 Función `pass` del agente.

Cuando se recibe una `get request` SNMP por parte del agente, éste llama a la secuencia de comandos `passtest` con la opción `-g`, seguida del OID de la MIB. Por tanto, el agente llamará a la secuencia de comandos:

```
/bin/sh /usr/local/passtest -g .1.3.6.1.4.1.2021.255
```

Cuando el agente recibe una `getnext-request`, se utiliza la opción `-n` en lugar de `-g`. Con una `set-request` SNMP, se suministrarán la opción `-s` más el tipo y el valor del objeto MIB. Estas opciones se suministran a la secuencia de comandos, de modo que ésta pueda tomar la acción correcta basada en las diferentes solicitudes.

La secuencia de comandos también es responsable de generar la salida apropiada para satisfacer la solicitud SNMP y los requerimientos del agente. Por ejemplo, cuando se envía una `getnext-request` SNMP al agente, debe estar presente el siguiente OID después del OID suministrado. El formato de la salida incluye la cadena OID, el tipo y el valor asociado. Así, si la secuencia de comandos es para retornar un valor de cadena (por ejemplo, `Hello World`) para el objeto `.1.3.6.1.4.1.2021.255.3`, entonces la secuencia de comandos deberá retornar lo siguiente:

```
.1.3.6.1.4.1.2021.255.3
string
Hello World
```

Cualquier dato que la secuencia de comandos quiera retornar, debe cumplir los requerimientos del agente especificando los tres componentes; OID, tipo y valor. Cuando se invoca la secuencia de comandos, el agente incorpora estas MIB en los datos. La secuencia de comandos puede producir cualquier información, pero debe usar uno de los tipos de objetos MIB existentes predefinidos. La tabla 7.3 lista estos tipos de objeto MIB, los cuales informan al agente cómo formatear los datos desde la secuencia de comandos.

Tabla 7.3 Tipos de objetos MIB de agente soportados.

Tipo OID	Descripción
`string`	Es un tipo de dato que representa cero o más octetos, donde cada octeto puede tomar cualquier valor desde `0` hasta `255`

Tabla 7.3 Tipos de objetos MIB de agente soportados. (*continuación*)

Tipo OID	Descripción
`integer`	Es un tipo de dato que representa un número cardinal, donde no se hace ninguna limitación al nivel de precisión que puede requerirse para representar el valor
`objectid`	Es un tipo de dato representativo de un objeto nombrado autoritariamente, que consiste en una secuencia de valores que especifican un árbol MIB
`timeticks`	Es un tipo de dato que representa un número entero no negativo que cuenta el tiempo en centésimas de segundo desde alguna época establecida. `timeticks` es como un `counter` en todos los demás aspectos
`ipaddress`	Es un tipo de dato que representa una `octet string` que tiene una longitud de cuatro bytes (32 bits), donde cada uno de los cuatro octetos se relaciona con uno de los cuatro bytes de una dirección estándar IP
`counter`	Es un tipo de dato que representa un entero no negativo que se incrementa hasta que alcanza un valor máximo y después reinicia en cero. Un counter es un `integer` que puede tomar un valor entre `0` y `4294967295`. Un `counter` tiene un valor de inicio no definido
`gauge`	Es un tipo de dato que representa un entero no negativo que puede crecer o decrecer y se disparará en un valor máximo. Un `gauge` es como un `counter` en todos los demás aspectos

Para un mayor entendimiento de las capacidades de **pass**, considérese la secuencia de comando `passtest` mostrada en el listado 7.1. Esta secuencia de comandos viene con la versión estándar de Linux y puede encontrarse en el directorio `/usr/local`. La secuencia de comandos se proporciona como un ejemplo de cómo escribir una secuencia de comandos shell (módulo)

para extender el agente Linux. Como resultado, ésta no contiene todo lo que debería ser incluido (como el usuario) en una secuencia de comandos más robusta, con calidad de producción.

La primera observación que puede hacerse acerca de esta secuencia de comandos es que cada una de las operaciones SNMP –`set-request`, `getnext-request` y `get-request`– son detectadas específicamente. Es decir, la secuencia de comandos verifica, con el fin de asegurar que se ha suministrado el argumento proporcionado al programa, y que es `-s` o `-n`. Recuérdese que el agente invoca la secuencia de comandos con el argumento de línea de comandos apropiado que corresponde a la operación SNMP apropiada. Éstas son destacadas por la instrucción `if` hallada en las líneas 8 y 13. Estas dos líneas verifican si el primer argumento del programa (`$1`) es igual a `s` o a `n`. Si ninguna de éstas es suministrada, la secuencia de comandos supone que `g` fue proporcionada.

Listado 7.1 Secuencia de comandos `passtest`.

```
 1 #!/bin/sh -f
 2
 3 PATH=$path:/bin:/usr/bin:/usr/ucb
 4
 5 PLACE=".1.3.6.1.4.1.2021.255"
 6 REQ="$2"
 7
 8 if [ "$1" = "-s" ]; then
 9   echo $* >> /tmp/passtest.log
10   exit 0
11 fi
12
13 if [ "$1" = "-n" ]; then
14   case "$REQ" in
15    $PLACE)     RET=$PLACE.1 ;;
16    $PLACE.1)   RET=$PLACE.2.1 ;;
17    $PLACE.2.1) RET=$PLACE.2.2 ;;
18    $PLACE.2.2) RET=$PLACE.3 ;;
19    $PLACE.3)   RET=$PLACE.4 ;;
20    $PLACE.4)   RET=$PLACE.5 ;;
21    $PLACE.5)   RET=$PLACE.6 ;;
22    *)          exit 0 ;;
23  esac
24 else
25  case "$REQ" in
26    $PLACE)     exit 0 ;;
27    *)          RET=$REQ ;;
28  esac
```

(*continúa*)

```
29 fi
30 echo "$RET"
31 case "$RET" in
32    $PLACE.1) echo "string"; echo "life tne universe
       and everything"; exit 0 ;;
33    $PLACE.2.1) echo "integer"; echo "42"; exit 0 ;;
34    $PLACE.2.2) ecno "objectid"; echo
        ".1.3.6.1.4.42.42.42"; exit 0 ;;
35    $PLACE.3) echo "timeticks"; echo "363136200";
         exit 0 ;;
36    $PLACE.4) echo "ipaddress"; echo "127.0.0.1" ;;
37    $PLACE.5) echo "counter"; echo "42"; exit 0 ;;
38    $PLACE.6) echo "gauge"; echo "42"; exit 0 ;;
39    *) echo "string"; echo "ack... $RET $REQ"; exit 0 ;;
40 esac
```

Cuando la secuencia de comandos detecta que se emitió una `getnext request` SNMP, ésta debe recuperar el siguiente objeto OID que es lexicográficamente más grande que el anterior. Como resultado, la secuencia de comandos utiliza una instrucción `case` para determinar cuál objeto está siendo recuperado y avanza el OID correspondiente, colocando el valor OID en la variable `RET`. La variable `RET` contendrá el objeto OID retornado. Supóngase, por el momento, que el agente quiere recuperar el objeto OID `.1.3.6.1.4.1.2021.255.3`. Entonces, la instrucción `case` asignará la variable `PLACE.4` a `RET`. La variable `PLACE.4` contiene el valor `.1.3.6.1.4.1.2021.255.4`, el cual es entonces asignado a `RET`. Después de esto, el programa baja a la línea `30`, donde envía los contenidos de `RET` a la salida estándar, usando el comando de shell **echo**. Esta línea cumple el primer requerimiento de la secuencia de comandos para enviar de vuelta la cadena OID al agente. A continuación, la secuencia de comandos detecta cuál OID se mostró recientemente, mediante el uso de otra instrucción `case`. En esta etapa, la variable `RET` contiene aún el objeto `.1.3.6.1.4.1.2021.255.4`, el cual es igual a `$PLACE.4` y obliga a que se ejecuten los comandos asociados en la línea `36`. En este caso, la secuencia de comandos muestra la cadena `ipaddress` y el valor `127.0.0.1`. La cadena de `ipaddress` satisface el segundo requerimiento del agente, el cual especifica el tipo de objeto OIB. El valor final `127.0.0.1` que muestra la secuencia de comandos es el valor asociado del objeto `.1.3.6.1.4.1.2021.255.4`.

Para soportar la `get-request` SNMP, la parte `else` del código se ejecuta comenzando en la línea `24`. En esta instrucción `case`, si el argumento `REQ` (el cual es la cadena OID) es igual a la variable

P_ACE, entonces la secuencia de comandos saldrá con estado 0. Esto sucede porque el OID suministrado por el agente para la get request es el inicio de la tabla OID, y ningún valor puede retornarse bajo esta condición. De otra manera, se asigna cualquier otro valor OID a la variable RET, y la secuencia de comandos procesa la solicitud como se indicó antes, comenzando en la línea 30.

Nótese que en la línea 8, para la set request SNMP, los argumentos (usando el $*) dados a la secuencia de comandos son dirigidos a un archivo de registro llamado /tmp/passtest.log. Eso quiere decir que cualquier operación SNMP set-request del agente simplemente registra la información y retorna un código de salida 0. Esto se debe a que el agente invoca la secuencia de comandos cada vez que se tiene acceso a la tabla OID. Como resultado, la secuencia de datos no implementa ningún mecanismo para capturar las operaciones set y las aplica a los datos que genera la secuencia de comandos. Si se fuera a implementar la característica set, la secuencia de comandos necesitaría una forma de alterar cualquiera de los objetos OID que fueran cambiados. Debido a que éste ya utiliza el archivo passtest.log como almacenamiento temporal, sólo necesita analizar gramáticamente este archivo y alterar los valores OID contenidos en la secuencia de comandos. Una vez se hace esto, la secuencia de comandos debería remover las entradas previamente analizadas en el archivo de registro, de manera que éstas no son consideradas erróneamente nuevas cuando se requiera otra operación set. Como se puede observar, la versión actual de la secuencia de comandos passtest no implementa permanentemente set-requests. Cuando se ejecuta una operación SNMP getnext-request en el agente usando el OID soportado por la secuencia de comandos passtest, aparecerá lo siguiente:

```
enterprises.ucdavis.255.1 = "life the universe and everything"
enterprises.ucdavis.255.2.1 = 42
enterprises.ucdavis.255.2.2 = OID: 42.42.42
enterprises.ucdavis.255.3 = Timeticks: (363136200) 42 days,
   0:42:42.00
enterprises.ucdavis.255.4 = IpAddress: 1.0.0.127
enterprises.ucdavis.255.5 = 42
enterprises.ucdavis.255.6 = Gauge: 42
```

El listado 7.2 muestra otro ejemplo del comando **pass**. En este programa, llamado maintain-message.pl, la operación SNMP *set-request* se usa como un disparador (*trigger*) para borrar los

contenidos del archivo de mensajes del sistema. Esta secuencia de comandos representa un ejemplo funcional de cómo puede ayudar la facilidad `pass` a desarrollar algunas tareas útiles de la administración del sistema. En este ejemplo, la secuencia de comandos ayuda a mantener el archivo `/var/log/messages` a través de varios sistemas.

Listado 7.2 Secuencia de comandos **Perl** (`maintain message.pl`) para mantener el archivo de mensajes.

```
1 #!/usr/bin/perl -w
2 ##############################################################
3 #
4 # Extensible Agent Script Example - Maintenance of the
      /var/adm/messages file. It
5 # provides a way to delete the contents of the file to zero bytes
      and also shows the
6 # the last 50 lines of the file.
7 #
8 # Steve Maxwell
9 # December '99
10 #
11 ##############################################################
12 #
13 $MESSAGESFILE   "/var/log/messages";
14 $OID_PATHBASE = ".1.3.6.1.4.1.2021.255";
15 $SET_OID      = ".1.3.6.1.4.1.2021.255.51";
16 $SET_OID_OBJ  = 51;
17 $DEF_LINES    = 50;
18 $LOG_NAME     = "agent log";
19 $LOG_PATH     = "/etc/snmp/";
20 $DATE         = `/bin/date '+Date: %m/%d/%y Time: %H:%M:%S'`;
21 $TAIL         = "/usr/bin/tail -n";
22 $LOGGING      = 1;
23 ##
24 ## Creates logfile. On success, return 0, else 1.
25 ##
26 sub CreatLogFile {
27
28          $lf = sprintf("%s%s", $LOG_PATH, $LOG_NAME);
29          unless (open (LFP, ">> $lf" ) ) {
30                print "Can't open log file: $lf\n";
31                exit 1;
32          }
33          return 0;
34 }
35 ##
```

(continúa)

```
36 ## Logs any string passed to it.
37 ##
38 sub Log {
39          if ( $LOGGING ) {
40                print LFP "$DATE $_[0]\n";
41          }
42 }
43 ##
44 ## Close open log file and exit program with supplied
45 ## error code.
46 ##
47 sub Leave {
48          close LFP;
49          exit $_[0];
50 }
51 ##
52 ## We need to purge the log file, but we need to make
53 ## sure the correct OID string is provided to us.
54 ##
55 sub SnmpSet {
56          ($oid, $type, $value) = @_;
57
58          ##
59          ## Make sure OID from agent matches are OID set object.
60          ## if so, then we know to execute the SET operation.
61          ##
62          if($oid eq $SET_OID) {
63                Log "SnmpSet: oid...$oid, type...$type, and
                        value...$value\n";
64          $comm = sprintf("%s %s %s", "/bin/cp", "/dev/null",
                $MESSAGESFILE);
65          Log "Executing... $comm";
66          $rc = system ($comm);
67          $stat = $rc/256;
68          if(! $stat ) {
69                Log "SnmpSet: Command executes successfully";
70                Leave 0;
71          }
72    }
73    Leave 1;
74 }
75 ##
76 ## Get the next OID and line from the messages
77 ## file.
78 ##
79 sub SnmpGetNext {
80    ($Reqoid, $master_oid) = @_;
81
82    @data = `$TAIL $DEF_LINES $MESSAGESFILE`;
```

(*continúa*)

```
83
84          if ($Reqoid eq $master_oid) {
85                Log "SnmpGetNext: OIDs are equal\n";
86                print "$Reqoid.0\n";
87                print "string\n";
88                print "$data[0]";
89                Log "SnmpGetNext: $Reqoid.0, string, $data[0]";
90                Leave 0;
91    } else {
92                $newstring = "";
93                $oidstring = $Reqoid;
94                (@oids) = split (/\./, $oidstring);
95                $count = $#oids-1;
96                for ( $i = 1; $i <= $count; $i++ ) {
97                      $newstring = sprintf("%s.%s", $newstring,
                            $oids[$i]);
98                }
99                $lastoid = $oids[$#oids];
100               $oid_int = sprintf( "%i", $lastoid);
101               $oid_int++;
102               print "$newstring.$oid_int\n";
103               print "string\n";
104               print "$data[$oid_int]";
105               Log "SnmpGetNext: $newstring.$oid_int, string,
                     $data[$oid_int]";
106         }
107         Leave 0;
108 }
109 ##
110 ## Get an individual line of the messages file
111 ##
112 sub SnmpGet {
113         $Reqoid = $_[0];
114
115         Log "SnmpGet: $Reqoid";
116         @lines = `$TAIL $DEF_LINES $MESSAGESFILE`;
117
118         (@oids) = split (/\./, $Reqoid);
119
120         $lastoid = $oids[$#oids];
121         if ( $lastoid == $SET_OID_OBJ ) {
122               print "$Reqoid\n";
123               print "integer\n";
124               print "0";
125               Leave 0;
126         }
127         if ( $lastoid > 0 && $lastoid < 50 ) {
128               print "$Reqoid\n";
129               print "string\n";
```

(*continúa*)

```
130                 print "$lines[$lastoid]";
131                 Log "SnmpGet: $Reqoid, string, $lines[$lastoid]";
132                 Leave 0;
133         } else {
134                 Leave 1;
135         }
136 }
137 ##
138 ## Main program starts here
139 ##
140
141 CreatLogFile;
142
143 ($arg1, $arg2, $arg3, $arg4) = @ARGV;
144
145 if ($arg1 eq "-s") {
146         SnmpSet $arg2, $arg3, $arg4;
147 } elsif ($arg1 eq "-n") {
148         SnmpGetNext $arg2, $OID_PATHBASE;
149 } else {
150         SnmpGet $arg2;
151 }
```

Este ejemplo puede lucir un poco complicado, debido al número de líneas, pero realmente el programa es muy sencillo y simple. Además, el programa de ejemplo, llamado `messages.pl`, se escribió usando el lenguaje Perl y se usó con la versión `5.005_03` del interpretador. La parte central del programa está entre las líneas `141` y `151`, con la porción principal separada por varias funciones. Como se puede observar, el programa utiliza tres funciones que son modeladas después de la forma en que el agente invoca la secuencia de comandos. Las funciones incluyen `SnmpSet`, la cual comienza en la línea `55`; `SnmpGetNext`, la cual empieza en la línea `79`, y `SnmpGet`, la cual empieza en la línea `112`. Cada una de ellas es llamada por el agente, dependiendo de la operación SNMP que éste recibe. Lo que es realmente nuevo en este ejemplo, comparado con la secuencia de comandos `passtest`, es que éste soporta la operación `set request` SNMP. En este caso, la operación set utilizada en un OID específico (`1.3.6.1.4.1.2021.255.51`) ocasionará que los contenidos del archivo `/var/log/messages` sean vaciados. El comando real se halla en la línea `66`. En esta línea, el comando **system** se invoca con la variable `$comm`. En Perl, el comando **system** enviará la cadena directamente al sistema operativo para su ejecución. Esta variable tiene asignada la cadena `/bin/cp/dev/null /var/log/messages`. El resto de la función simplemente verifica el

estado de retorno del comando `system`, y registra la actividad de la secuencia de comandos. Si el comando es exitoso, la secuencia de comandos se termina con un valor de éxito; de otra manera, ésta retorna un estado de error (línea 73). Una función llamada `Leave` se define para cerrar el archivo de registro y retornar la condición de estado correcta antes de salir de la secuencia de comandos. Esta función se define en las líneas 47 y 50.

La secuencia de comandos `messages.pl` también proporciona una forma de ver hasta 50 líneas del archivo de mensajes. Para tener acceso a los contenidos de este archivo se hace un `getnext-request` en el objeto MIB `1.3.6.1.4.1.2021.255.1`. Recuérdese que la operación `getnext` recorrerá un árbol MIB para recuperar toda la MIB o un grupo de objetos MIB. Por tanto, se pueden obtener todos los objetos bajo el árbol `255`. Como consecuencia, la salida del ejemplo se producirá como se ve en el listado 7.3. Las cadenas entre comillas representan las líneas reales del archivo `/var/log/messages`.

Listado 7.3 Un "recorrido" SNMP del archivo `/var/log/messages`.

```
enterprises.ucdavis.255.0   "Sep 1 20:04:30 socrates chat[995]: expect (OK)"
enterprises.ucdavis.255.1 = "Sep 1 20:05:15 socrates pppd[993]: Connect script failed"
enterprises.ucdavis.255.2 = "Sep 1 20:05:15 socrates chat[995]: alarm"
enterprises.ucdavis.255.3 = "Sep 1 20:05:15 socrates chat[995]: Failed"
enterprises.ucdavis.255.4 = "Sep 1 20:05:16 socrates pppd[993]: Exit."
enterprises.ucdavis.255.5 = "Sep 1 20:05:16 socrates ifup ppp: pppd started for ppp0 on /dev/modem
   at 115200"
enterprises.ucdavis.255.6 = "Sep 1 20:05:16 socrates pppd[1000]: pppd 2.3.7 started by root, uid 0"
enterprises.ucdavis.255.7 = "Sep 1 20:05:17 socrates chat[1002]: abort on (BUSY)
enterprises.ucdavis.255.8 = "Sep 1 20:05:17 socrates chat[1002]: abort on (ERROR)"
enterprises.ucdavis.255.9 = "Sep 1 20:05:17 socrates chat[1002]: abort on (NO CARRIER)"
enterprises.ucdavis.255.10 = "Sep 1 20:05:17 socrates chat[1002]: abort on (NODIALTONE)"
enterprises.ucdavis.255.11 = "Sep 1 20:05:17 socrates chat[1002]: abort on (Invalid Login)"
enterprises.ucdavis.255.12 = "Sep 1 20:05:17 socrates chat[1002]: abort on (Login incorrect)"
enterprises.ucdavis.255.13 = "Sep 1 20:05:17 socrates chat[1002]: send (ATZ^M)"
enterprises.ucdavis.255.14 = "Sep 1 20:05:17 socrates chat[1002]: expect (OK)»
enterprises.ucdavis.255.15 = "Sep 1 20:06:02 socrates pppd[1000]: Connect script failed"
enterprises.ucdavis.255.16 = "Sep 1 20:06:02 socrates chat[1002]: alarm"
enterprises.ucdavis.255.17 = "Sep 1 20:06:02 socrates chat[1002]: Failed"
enterprises.ucdavis.255.18 = "Sep 1 20:06:03 socrates pppd[1000]: Exit."
enterprises.ucdavis.255.19 = "Sep 1 20:06:04 socrates pppd[1011]: pppd 2.3.7 st
   arted by root, uid 0"
enterprises.ucdavis.255.20 = "Sep 1 20:06:04 socrates ifup-ppp: pppd started for ppp0 on /dev/modem
   at 115200"
enterprises.ucdavis.255.21 = "Sep 1 20:06:05 socrates chat[1012]: abort on (BUSY)"
enterprises.ucdavis.255.22 = "Sep 1 20:06:05 socrates chat[1012]: abort on (ERROR)"
enterprises.ucdavis.255.23 = "Sep 1 20:06:05 socrates chat[1012]: abort on (NOCARRIER)"
enterprises.ucdavis.255.24 = "Sep 1 20:06:05 socrates chat[1012]: abort on (NODIALTONE)"
enterprises.ucdavis.255.25 = "Sep 1 20:06:05 socrates chat[1012]: abort on (Invalid Login)"
enterprises.ucdavis.255.26 = "Sep 1 20:06:05 socrates chat[1012]: abort on (Login incorrect)"
enterprises.ucdavis.255.27 = "Sep 1 20:06:05 socrates chat[1012]: send (ATZ^M)"
enterprises.ucdavis.255.28 = "Sep 1 20:06:05 socrates chat[1012]: expect (OK)"
enterprises.ucdavis.255.29 = "Sep 1 20:06:50 socrates chat[1012]: alarm"
enterprises.ucdavis.255.30 = "Sep 1 20:06:50 socrates chat[1012]: Failed"
enterprises.ucdavis.255.31 = "Sep 1 20:06:50 socrates pppd[1011]: Connect scrip
   t failed"
```

(*continúa*)

```
enterprises.ucdavis.255.32 =  Sep 1 20:06:51 socrates pppd[1011]: Exit."
enterprises.ucdavis.255.33 =  Sep 1 20:06:51 socrates pppd[1024]: pppd 2.3 7 st
arted by root, uid 0"
enterprises.ucdavis.255.34 = "Sep 1 20:06:51 socrates ifup ppp: pppd started fo
r ppp0 on /dev/modem at 115200"
enterprises.ucdavis.255.35 =  "Sep 1 20:06:52 socrates chat[1025]: abort on (BUSY)"
enterprises.ucdavis.255.36 = "Sep 1 20:06.52 socrates chat[1025]: abort on (ERROR)"
enterprises.ucdavis.255.37 = "Sep 1 20:06:52 socrates chat[1025]: abort on (NO CARRIER)"
enterprises.ucdavis.255.38 = "Sep 1 20:06.52 socrates chat[1025]: abort on (NO DIALTONE)"
enterprises.ucdavis.255.39 =  Sep 1 20.06:52 socrates chat[1025]: abort on (Invalid Login)"
enterprises.ucdavis.255.40 = "Sep 1 20:06:52 socrates chat[1025]: abort on (Login Incorrect)"
enterprises.ucdavis.255.41 =  "Sep 1 20:06:52 socrates chat[1025]: send (ATZ^M)"
enterprises.ucdavis.255.42 = "Sep 1 20:06:52 socrates chat[1025]: expect (OK)"
enterprises.ucdavis.255.43 = "Nov 2 20:01:02 socrates last message repeated 2 times"
enterprises.ucdavis.255.44 = "Nov 2 20:01:03 socrates PAM_pwdb[914]: (su) session opened for user
   news by (uid=0)"
enterprises.ucdavis.255.45 = "Nov 2 20:01.06 socrates PAM_pwdb[914]: (su) session closed for user
   news"
enterprises.ucdavis.255.46 = "Nov 2 20:01:15 socrates snmpd: Connection from 10.0.2.201"
enterprises.ucdavis.255.47 = "Nov 2 20:04:50 socrates last message repeated 2 times"
enterprises.ucdavis.255.48 = "Nov 2 20:06:16 socrates last message repeated 3 times"
enterprises.ucdavis.255.49 = "Nov 2 20:11:40 socrates last message repeated 102 times"
```

La salida en el listado 7.3 corresponde a las últimas 50 líneas del archivo de mensajes del sistema. Como se observa en la secuencia de comandos, las líneas del archivo `messages` se obtienen de la función `SnmpGetNext` que comienza en la línea 79. La línea 82 contiene el comando UNIX **tail**. Aquí se invoca este comando y la salida se almacena en un arreglo (o lista) variable llamado `data`. Cuando se invoca esta función, una de las líneas del arreglo se muestra en la salida estándar; esto corresponde al objeto OID que fue referenciado con la llamada SNMP. Cuando se emite un recorrido SNMP, llama a esta función múltiples veces con la OID del siguiente objeto en el árbol. Las líneas que van de la 92 a la 105 examinan el último número del OID y lo incrementan en uno (línea 101). A continuación, la función muestra el nuevo OID, el tipo de cadena y la salida de una línea desde el comando **tail** usando el mismo OID como un índice dentro del arreglo. La función `SnmpGet` opera básicamente de la misma forma; ésta ejecuta el comando UNIX **tail** y retorna una de las líneas desde el archivo de mensaje usando el OID como un índice en el arreglo `data`.

Puede hacerse un par de observaciones acerca de esta secuencia de comandos. Primero, la función `SnmpSet` puede ser más robusta respecto del vaciado del archivo `messages`. Ésta podría, por ejemplo, definir otro objeto que almacenaría la información de fecha y hora de cuando se realizó el vaciado. Ésta, ciertamente, sería una característica útil en un gran entorno de red. A continuación, como los elementos del arreglo en PERL inician en cero, la salida del comando **tail** no empata exactamente con los números OID. Observen en la salida de arriba las líneas que inician en

1.3.6.1.4.1.2021.255.0 frente a 1.3.6.1.4.1.2021.255.1. Por último, la secuencia de comandos podría modificarse para manejar límites fuertemente codificados y otros datos relevantes de una manera diferente. Por ejemplo, podría verificar para determinar el rango del OID enviado a éste por el agente. Esto se muestra comenzando en la línea 127. Un programa sofisticado usaría un enfoque más dinámico cuando se determina el rango de la OID, etc.

Archivo ejemplo de configuración de agente

El listado 7.4 muestra un archivo ejemplo de configuración del agente. Éste podría ser un buen punto de partida para quienes quieran emplear el entorno del agente cuando es crítico hacer un seguimiento de algunos de los procesos UNIX estándar. Además, la configuración incluye funciones de llamado para limitar automáticamente ciertos procesos del sistema, de modo que éste no llega a sobrecargarse.

Listado 7.4 Ejemplo de configuración del agente.

```
#
# Sample Configuration
# Monitor these processes..
#
proc mountd
proc nfsd
proc lpsched
#
# Monitor load average above 2.0..
#
load 2
#
# Make sure we have a least 50 MB in root
# and 100 MB in /usr
#
disk / 50000
disk /usr 100000
#
# Execute the following programs to obtain
# output via the UCD mib...
#
exec .1.3.6.1.4.1.2021.255.1 top /usr/bin/top
exec .1.3.6.1.4.1.2021.255.2 ps /bin/ps -e
pass .1.3.6.1.4.1.2021.255.3 /usr/local/perl /etc/snmp/message.pl
```

Opciones de línea de comandos

El agente proporciona varias opciones de línea de comandos, como se ve en la tabla 7.4. Debido a que el agente, en muchos casos, será arrancado automáticamente cuando el sistema inicie,

algunas de estas opciones deberían cambiarse en el archivo de inicio, si van ser permanentes. Sin embargo, algunas de las opciones sólo tienen sentido cuando se usan en la línea de comandos.

Cuando se arranca el `snmpd`, éste utiliza el bien conocido puerto `161` SNMP de manera predeterminada. Éste puerto debe cambiarse si otros agentes se ejecutan en el mismo sistema o si se está usando un agente maestro. La opción `-p` se suministra para este propósito. El comando `snmpd` usará esta opción, seguida del número de puerto, para escuchar solicitudes entrantes SNMP.

Varias opciones pueden utilizarse para ayudar en cualquiera de los esfuerzos de depuración que deban hacerse con el agente. La opción `-a` muestra la dirección IP de cualquier sistema que haya estado en comunicación con el agente, junto con la información de fecha y hora. Esto es útil para establecer que cualquiera de las ACL configuradas están trabajando apropiadamente. Cuando se utiliza una opción, el agente mostrará mensajes como los siguientes:

```
1999 11 21 16:04:17 Received SNMP packet(s) 10.0.2.126
```

La opción `-d` muestra los paquetes UDP enviados y recibidos desde el agente, y se utiliza para ayudar a detectar problemas relacionados con el protocolo SNMP u otros aspectos de las operaciones. Además, esta opción puede utilizarse para examinar el objeto MIB solicitado desde las aplicaciones clientes SNMP. Un ejemplo de la salida de depuración del agente se muestra en seguida:

```
GET
   -- system.sysContact.0
   >> system.sysContact.0 = Steve Maxwell
sent 64 bytes to 10.0.2.126:
0000: 30 82 00 3C 02 01 00 04 06 70 75 62 6C 69 63 A2 0..<.....public.
0016: 82 00 2D 02 04 23 C1 E0 9F 02 01 00 02 01 00 30 ..-..#.........0
0032: 82 00 1D 30 82 00 19 06 08 2B 06 01 02 01 01 04 ...0.....+......
0048: 00 04 0D 53 74 65 76 65 20 4D 61 78 77 65 6C 6C ...Steve Maxwell
```

En este ejemplo, el agente obtiene una solicitud `get` SNMP desde el sistema con dirección IP `10.0.2.126`. El agente muestra la cadena " " para indicar lo que la entidad remota SNMP solicita (en este caso, el objeto `sysContact`) y la cadena ">>" muestra lo que el agente retorna. La salida hexadecimal corresponde a todo el paquete UDP, el cual incluye la información de la cadena comunidad y cualquier objeto(s) retornado. Si se quiere

sólo la transacción de protocolo y no la pista del paquete UDP, se usa la opción `-V`. Esta opción mostrará

```
GET
   -- system.sysLocation.0
   >> system.sysLocation.0 = Graphics Lab 11 B
```

como un resultado de una solicitud para el objeto `sysLocation` MIB desde una aplicación cliente SNMP.

La opción `-q`, de acuerdo con la documentación del paquete UCD, muestra un formateo más simple que puede usarse, por ejemplo, para automatizar el análisis sintáctico de la salida del agente. Desafortunadamente, esta opción no parece hacer nada en la versión 4 del agente probado.

Para mayor ayuda con la depuración del agente, la opción `-f` instruye al agente a no bifurcarse (*fork*) del shell. Sin esta opción, el agente será separado por bifurcación (*fork*) en un nuevo proceso desde el shell actual y retornará el indicador de comandos del shell. El agente `mibiisa`, por ejemplo, se comporta de esta manera. Cuando se utiliza la opción `-f`, el agente se puede terminar presionando **^c** en el shell que fue utilizado para arrancarlo. Como otra alternativa para detener el agente, se utiliza el comando `kill` con la identificación apropiada del proceso del agente.

Si es necesario grabar la información de depuración en un archivo, puede utilizarse la opción `-l` seguida del nombre de un archivo. La salida registrada en este archivo incluye las salidas estándar (`STDOUT`) y error estándar (`STDERR`). Si la opción `-l` se utiliza sin un nombre de archivo, entonces se utiliza el archivo predefinido `/var/log/snmpd.log`. Cuando se conduce una pista SNMP (con la opción `-V`), guardar la salida en un archivo será útil cuando se aborda una gran cantidad de transacciones. Sin embargo, si se quiere registrar información dirigida a las salidas estándar y error estándar, se utiliza entonces la opción `-L`. Las opciones `-L` y `-l`, son mutuamente excluyentes y, por tanto, no deberán usarse de manera conjunta.

Si se quiere que el agente utilice un archivo de configuración específico, se usa la opción `-c` seguida de la ruta completa y el nombre del archivo. De manera predeterminada, el agente busca en ubicaciones preconfiguradas uno o más archivos de configuración del agente. Si no se quiere que el agente busque archivos adicionales de configuración y, en cambio, se quiere usar solamente el especificado con la opción `-c`, se incluye también la opción `-C`. Esto obliga al agente a usar sólo el archivo de configuración que

se proporcionó, y es una de las formas de asegurar que se ignore cualquier otro archivo de configuración.

MIB Linux/UCD

El agente implementa objetos MIB bajo el número empresarial `ucdavis (2021)`. La ruta completa para el árbol UCD empieza con `iso.org.dod.internet.private.enterprises` o `1.3.6.1.4.1`. El UCD MIB proporciona los siguientes objetos y grupos importantes MIB:

- `procTable`
- `memory`
- `extTable`
- `diskTable`
- `loadTable`
- `systemStats`
- `version`
- `snmpErrors`

Esta definición MIB puede hallarse en el archivo `/usr/local/share/snmp/mibs/UCD-SNMP-MIB.txt` después de instalar el paquete UCD.

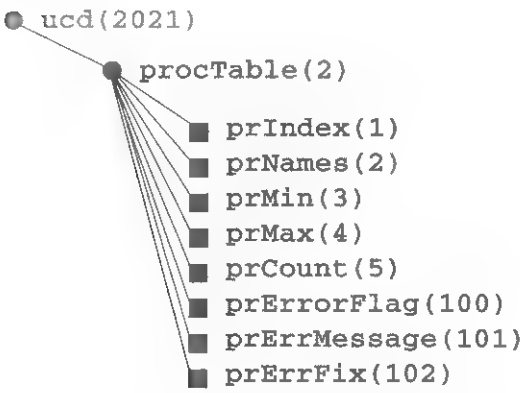


Figura 7.3 Vista de árbol de `procTable`.

ProcTable UCD

La tabla `procTable` consta de objetos que obtienen una lista de procesos del sistema que son seguidos por el agente y han sido colocados en el archivo de configuración del agente. La figura 7.3

muestra una representación jerárquica de los objetos, y cómo se relacionan con la rama ucdavis. A la información del proceso se tiene acceso a través del objeto MIB prEntry. La tabla procTable contiene el siguiente conjunto de objetos:

Nombre del objeto:	prIndex
OID:	prEntry.1
Tipo de objeto:	PRIndex (Integer)
Modo de acceso:	sólo lectura
Descripción:	Índice de referencia para cada proceso del sistema al que se le hace seguimiento. Debido a que procTable es en realidad una tabla, el prIndex se utiliza para tener acceso a otra instancia de la tabla

Nombre del objeto:	prNames
OID:	prEntry.2
Tipo de objeto:	Display String [255]
Modo de acceso:	sólo lectura
Descripción:	Nombre del proceso al que el agente está haciendo seguimiento. El nombre del proceso se obtiene del archivo de configuración del agente después del comando de la palabra clave **proc**

Nombre del objeto:	prMin
OID:	prEntry.3
Tipo de objeto:	Integer
Modo de acceso:	sólo lectura
Descripción:	Indica el número mínimo de procesos que deberían ser ejecutados en el sistema. Si el número de procesos es menor que el mínimo, se genera una bandera de error en los objetos MIB prErrorFlag y prErr-Message. El

valor mínimo se obtiene de la configuración del agente después del valor máximo, si es que se ha especificado alguno

Nombre del objeto: prMax
OID: prEntry.4
Tipo de objeto: Integer
Modo de acceso: sólo lectura
Descripción: Indica el número máximo de procesos que deberían ser ejecutados en el sistema. Si el número de procesos es mucho más grande que el máximo, se generará una bandera de error en otros objetos MIB. El valor máximo se obtiene de la configuración del agente después de especificar el nombre del proceso

Nombre del objeto: prCount
OID: prEntry.5
Tipo de objeto: Integer
Modo de acceso: sólo lectura
Descripción: Especifica el número actual de procesos corriendo en el sistema en este momento, a los que el agente está haciendo seguimiento

Nombre del objeto: prErrorFlag
OID: prEntry.100
Tipo de objeto: Integer
Modo de acceso: sólo lectura
Descripción: Este objeto se establece en 1 si el agente detecta problemas con el

proceso, y se establece en 0 si no se detecta ningún problema. Entre los posibles problemas están el acceso de la cuenta de proceso máximo y mínimo o que el proceso no está corriendo

Nombre del objeto:	`prErrMessage`
OID:	`prEntry.101`
Tipo de objeto:	`DisplayString [255]`
Modo de acceso:	`sólo lectura`
Descripción:	Si el agente detecta un problema con un proceso, este objeto contendrá un mensaje de error que describe el problema con más detalle

Nombre del objeto:	`prErrFix`
OID:	`prEntry.102`
Tipo de objeto:	`Integer`
Modo de acceso:	`sólo lectura`
Descripción:	Si el agente está configurado para soportar una respuesta al problema encontrado con un proceso, fijando este objeto en 1 instruirá al agente para que intente arreglar el problema del proceso automáticamente por medio del proceso que conoce el agente o de una secuencia de comandos

Tabla del agente extensible UCD

La tabla `extTable` consta de objetos que obtienen una lista de secuencia de comandos o programas que el agente ejecutará y cuyos resultados almacenará en la MIB UCD. La figura 7.4 muestra una representación jerárquica de estos objetos, y cómo se relacionan con la rama `ucdavis`. Esta tabla contiene un objeto

MIB llamado `extEntry`, el cual se utiliza para tener acceso a cada elemento `extTable`.

La tabla `extTable` contiene el siguiente conjunto de objetos:

Nombre del objeto:	`extIndex`
OID:	`extEntry.1`
Tipo de objeto:	`ExtIndex (Integer)`
Modo de acceso:	`sólo lectura`
Descripción:	Es el índice de la tabla `extTable` para cada secuencia de comandos extensible configurada

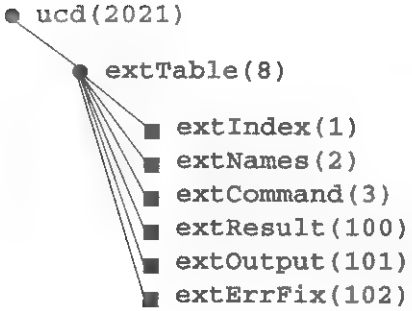


Figura 7.4 Vista de `extTable`.

Nombre del objeto:	`extNames`
OID:	`extIndex.2`
Tipo de objeto:	`DisplayString [255]`
Modo de acceso:	`sólo lectura`
Descripción:	Presenta una breve descripción (una palabra) del comando extensible especificado en el archivo de configuración del agente

Nombre del objeto:	`extCommand`
OID:	`extIndex.3`

Tipo de objeto:	`DisplayString [255]`
Modo de acceso:	`sólo lectura`
Descripción:	Línea de comandos extensible especificada en el archivo de configuración del agente

Nombre del objeto:	`extResult`
OID:	`extIndex.100`
Tipo de objeto:	`Integer`
Modo de acceso:	`sólo lectura`
Descripción:	Código de retorno (código de salida) del comando extensible ejecutado por el agente

Nombre del objeto:	`extOutput`
OID:	`extIndex.101`
Tipo de objeto:	`DisplayString [255]`
Modo de acceso:	`sólo lectura`
Descripción:	Primera línea de salida del comando extensible ejecutado por el agente

Nombre del objeto:	`extErrFix`
OID:	`extIndex.102`
Tipo de objeto:	`DisplayString [255]`
Modo de acceso:	`sólo lectura`
Descripción:	Bandera para controlar si el agente debiera intentar una mejoría del problema encontrado cuando el comando extensible está corriendo

Tabla de disco UCD

La tabla `diskTable` consta de objetos que pueden usarse para obtener una lista de todos los sistemas de archivo a los que el agente

está haciendo seguimiento, y de cualquiera de los objetos relacionados. La figura 7.5 muestra una representación jerárquica de estos objetos y cómo se relacionan con la rama `ucdavis`. Esta tabla contiene un objeto MIB llamado `dskEntry`, el cual se utiliza para tener acceso a cada elemento `diskTable`.

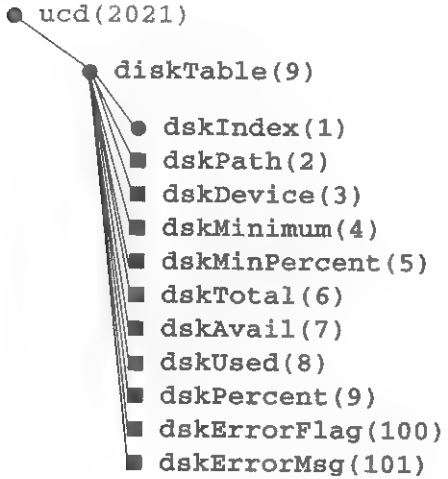


Figura 7.5 Vista de árbol de `diskTable`.

La tabla `diskTable` contiene el siguiente conjunto de objetos:

Nombre del objeto: `dskIndex`
OID: `dskEntry.1`
Tipo de objeto: `DiskIndex (Integer)`
Modo de acceso: `sólo lectura`
Descripción: Es un índice dentro de la tabla `diskTable` para cada sistema de archivo seguido o monitoreado por el agente

Nombre del objeto: `dskPath`
OID: `dskEntry.2`
Tipo de objeto: `DisplayString [255]`

Modo de acceso:	`sólo lectura`
Descripción:	Punto de montaje del sistema de archivo seguido o monitoreado por el agente

Nombre del objeto:	`dskDevice`
OID:	`dskEntry.3`
Tipo de objeto:	`DisplayString [255]`
Modo de acceso:	`sólo lectura`
Descripción:	La participación (*slice*) actual del dispositivo del disco montado

Nombre del objeto:	`dskMinimum`
OID:	`dskEntry.3`
Tipo de objeto:	`Integer`
Modo de acceso:	`sólo lectura`
Descripción:	Cantidad mínima de espacio del disco requerida, en kilobytes, antes de que el agente dispare un evento. Este valor se especifica con este objeto o con el objeto `dskMinPercent` del archivo de configuración del agente

Nombre del objeto:	`dskMinPercent`
OID:	`dskEntry.5`
Tipo de objeto:	`Integer`
Modo de acceso:	`sólo lectura`
Descripción:	Cantidad mínima de espacio del disco requerida, en kilobytes, antes de que el agente dispare un evento. Este valor se especifica con este objeto o con el objeto `dskMinimum` del archivo de configuración del agente

Nombre del objeto:	`dskTotal`
OID:	`dskEntry.6`
Tipo de objeto:	`Integer`
Modo de acceso:	`sólo lectura`
Descripción:	Tamaño total del disco/partición, en kilobytes

Nombre del objeto:	`dskAvail`
OID:	`dskEntry.7`
Tipo de objeto:	`Integer`
Modo de acceso:	`sólo lectura`
Descripción:	Cantidad total de espacio del disco disponible para el disco/partición, en kilobytes

Nombre del objeto:	`dskUsed`
OID:	`dskEntry.8`
Tipo de objeto:	`Integer`
Modo de acceso:	`sólo lectura`
Descripción:	Cantidad de espacio utilizada en el disco/partición, en kilobytes.

Nombre del objeto:	`dskPercent`
OID:	`dskEntry.9`
Tipo de objeto:	`Integer`
Modo de acceso:	`sólo lectura`
Descripción:	Cantidad de espacio utilizada en el disco/partición, como un porcentaje

Nombre del objeto:	`dskErrorFlag`
OID:	`dskEntry.100`

Tipo de objeto:	`Integer`
Modo de acceso:	`sólo lectura`
Descripción:	Se emplea para indicar que el disco/partición está por debajo de la cantidad mínima de espacio libre del disco especificada por el archivo de configuración del agente

Nombre del objeto:	`dskErrorMsg`
OID:	`dskEntry.101`
Tipo de objeto:	`DisplayString [255]`
Modo de acceso:	`sólo lectura`
Descripción:	Mensaje descriptivo de advertencia y resumen del espacio libre en relación con el disco/partición al que se le hace seguimiento

Tabla de carga del sistema UCD

La tabla `loadTable` consta de objetos que hacen un seguimiento a la carga del sistema. La figura 7.6 muestra una representación jerárquica de estos objetos y cómo se relacionan con la rama `ucdavis`. Esta tabla contiene un objeto MIB llamado `laEntry`, el cual es utilizado para tener acceso cada elemento `loadTable`.

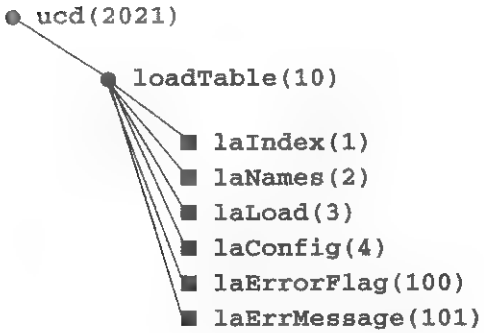


Figura 7.6 Vista de árbol de `loadTable`.

La tabla `loadTable` contiene el siguiente conjunto de objetos:

Nombre del objeto:	`laIndex`
OID:	`laEntry.1`
Tipo de objeto:	`Integer`
Modo de acceso:	`sólo lectura`
Descripción:	Es un índice dentro de la tabla `loadTable` para cada entrada

Nombre del objeto:	`laNames`
OID:	`laEntry.2`
Tipo de objeto:	`Display String [255]`
Modo de acceso:	`sólo lectura`
Descripción:	Lista de los nombres de carga promedio a los que el agente está haciendo seguimiento. Ésta puede ser una de las tres ventanas de carga promedio que el agente soporta: uno, cinco y 15 minutos. Éstas se representan mediante las cadenas `Load-1`, `Load-5` y `Load-15`, respectivamente

Nombre del objeto:	`laLoad`
OID:	`laEntry.3`
Tipo de objeto:	`Display String [255]`
Modo de acceso:	`sólo lectura`
Descripción:	Promedios de carga del sistema actuales para uno, cinco y 15 minutos, registrados por el agente

Nombre del objeto:	`laConfig`
OID:	`laEntry.4`

Tipo de objeto:	`Display String [255]`
Modo de acceso:	`sólo lectura`
Descripción:	Contiene el umbral de vigilancia para el promedio de carga, con el fin de señalar un error. Este objeto obtiene su valor del archivo de configuración del agente utilizando el comando **load**

Nombre del objeto:	`laErrorFlag`
OID:	`laEntry.100`
Tipo de objeto:	`Integer`
Modo de acceso:	`sólo lectura`
Descripción:	Se utiliza para indicar si se ha excedido el umbral de promedio de carga (contenido en el objeto `laConfig`). Si el valor es `1`, entonces se ha excedido el umbral; de otra forma, el valor es `0`

Nombre del objeto:	`laErrMessage`
OID:	`laEntry.101`
Tipo de objeto:	`Display String [255]`
Modo de acceso:	`sólo lectura`
Descripción:	Es un mensaje descriptivo de error que indica el promedio de carga y el valor del exceso en relación con el umbral

Agente maestro de Sun

Configuración del agente maestro/configuración del subagente

Antes de utilizar un subagente, éste debe registrarse con el agente maestro. Esto se logra mediante la actualización de un pequeño número de archivos de configuración que están localizados en el directorio `/etc/snmp/conf`. Cuando el agente maestro inicia,

analiza gramaticalmente cualesquier registro y archivos de recursos localizados en este directorio. El agente maestro también se asegura de que los mensajes fluyan desde la red hasta el subagente correcto, y de que los mensajes emitidos desde los subagentes sean también pasados correctamente. Para hacer el registro con el agente maestro `snmpdx`, los siguientes archivos deben contener entradas para el subagente:

```
agent.reg
agent.rsrc
```

El prefijo `agent` representa el nombre del subagente que se configurará bajo el agente maestro. El archivo `agent.reg` registra al subagente con el agente maestro. El archivo de registro consta de una palabra clave (o señal) seguida de un solo valor, separado por una nueva línea. En estos archivos se permiten los comentarios si se utiliza el carácter número (#). Una colección sencilla de palabras clave y valores se encierran en corchetes y constituyen una entrada de un subagente. Las palabras clave que pueden utilizarse en el archivo de registro incluyen **name**, **subtrees**, **time-out** y **watch-dog-time**.

La palabra clave **name** es sólo una etiqueta para nombrar al subagente internamente. La palabra clave **subtrees** especifica el árbol MIB que el subagente soporta cuando es consultado por otra entidad SNMP. La palabra clave **timeout** controla qué tanto tiempo el agente maestro debería esperar al subagente para que procese la solicitud SNMP. El valor `timeout` se expresa en microsegundos. La palabra clave **watch-dog-time** la utiliza el agente maestro para determinar si el subagente está aún funcionando, por medio de una verificación periódica del estado del subagente. El archivo de registro, mostrado en el listado 7.5, que usa el subagente `mibiisa` ilustra más este hecho.

Listado 7.5 Archivo de registro del agente.

```
agents =

        {
                name = "snmpd"
                subtrees = { mib-2, sun }
                timeout = 2000000
                watch-dog-time = 86400

}
```

En este caso, el nombre del subagente `mibiisa` es `snmpd`, el cual es utilizado internamente por el agente maestro. Los valores `subtrees` denotan dos macros que son abreviadas para cadenas específicas OID. El valor `mib 2` se define como `1.3.6.1.2.1`, y `sun` es `1.3.6.1.4.1.42`. El valor `timeout` es 2 segundos (o 2,000,000 microsegundos); el valor `watch-dog time` se expresa en segundos.

El archivo de recurso, `agent.rsrc`, se necesita también cuando se configura un subagente para trabajar con un agente maestro. Este archivo se usa principalmente para indicar el archivo de registro y el nombre del programa del subagente. El listado 7.6 presenta el archivo de recurso para el subagente `mibiisa`. Este archivo comparte el mismo formato que el archivo de registro, con las palabras clave seguidas de valores. El archivo de recurso requiere de las palabras clave **registration_file**, **policy**, **type** y **command**, y valores asociados. La señal **registration_file** asigna el archivo de registro del subagente que utilizará el agente maestro. La señal **policy** se utiliza para indicar cómo debería ejecutar el agente maestro al subagente. La señal **command** representa al programa del subagente que el agente maestro arrancará después de que éste complete el análisis gramatical, tanto de los archivos de registro como de recurso de cada uno de los subagentes. El puerto SNMP también puede especificarse en este archivo por medio del uso de la palabra clave **port**. Sin embargo, el subagente también puede permitir que el agente maestro asigne un puerto para usar por medio de la cadena `$PORT`. En el listado 7.6, el subagente `mibiisa` utiliza la cadena `$PORT`, en lugar de asignar estáticamente un puerto.

Listado 7.6 Archivo de recurso del agente.

```
resource =
|
        {
            registration_file = "/etc/snmp/conf/mibiisa.reg"
            policy = "spawn"
            type = "legacy"
            command = "/usr/lib/snmp/mibiisa -p $PORT"
        }
|
```

El agente maestro requiere los siguientes archivos:

```
snmpdx.acl
snmpdx.reg
snmpdx.rsrc
```

El archivo `snmpdx.acl` puede utilizarse para especificar la información de la cadena de comunidad que el agente maestro utilizará

para controlar el acceso a sus MIB. El mecanismo de control lo utilizan principalmente los objetos MIB del agente maestro, no las MIB del subagente. Ésta es una importante distinción porque los subagentes (por ejemplo, el agente `mibiisa`) cuentan con el archivo `snmpd.conf`, pero no con el archivo `snmpdx.acl`, para este propósito. El archivo `snmpdx.reg` es el archivo de registro del agente maestro que sirve para los mismos propósitos que el archivo del subagente. El archivo `snmpdx.rsrc` se utiliza para especificar algunos parámetros de tiempo para el agente maestro. Todos estos archivos están localizados en el directorio `/etc/snmp/conf`.

Opciones de línea de comandos

El agente `snmpdx` soporta varias opciones de línea de comandos, las cuales se listan en la tabla 7.4 y se describen en la siguiente sección.

Tabla 7.4 Opciones de línea de comandos del agente maestro.

Opción	Descripción
`-a`	Especifica la ruta del archivo de control de acceso utilizada por el agente
`-c`	Cambia la ruta en la cual se busca al archivo de configuración del agente
`-d`	Activa el nivel de depuración
`-h`	Ayuda de impresión en línea
`-i`	Especifica la ruta de otro archivo de estado utilizado por un agente
`-m`	Instruye al agente para pasar (*forward*) mensajes SNMP basados en un modo especificado
`-o`	Especifica la ruta del archivo de transformación empresarial OID
`-p`	Especifica el puerto que se usa, en lugar del puerto predeterminado `161`
`-r`	Especifica la ruta del archivo de recurso
`-y`	Invoca la recuperación de los subagentes

La opción `-a` especifica la ruta y el nombre del archivo para el archivo de control de acceso. El agente utiliza este archivo para controlar el acceso a la MIB por entidades externas. El archivo predeterminado es `/etc/snmp/conf/snmpdx.acl`. La opción `-i` da otra ruta completa para el archivo de estado del agente maestro. Éste es el archivo utilizado por el agente maestro para recuperación después de una caída. Contiene la identificación UNIX del proceso, número de puerto, nombre de recurso y nombre del agente de cada subagente definido. La localización predeterminada y el nombre del archivo de estado es `/var/snmp/snmpdx.st`.

La opción `-c` cambia la configuración predeterminada de la ruta desde `/etc/snmp/conf`. El agente maestro busca en esta localización la configuración del agente, el archivo de recurso y los archivos de transferencia OID. La opción `-o` de la línea de comandos cambia la localización predeterminada y el nombre del archivo OID del nombre de la empresa. El agente maestro utiliza este archivo para transformar el nombre de los vendedores en números de empresa. Este archivo es, de manera predeterminada, `/etc/snmp/conf/enterprises.oid`. El proceso Trap, por medio del agente maestro, requiere esta información para determinar información específica del vendedor. La opción `-r` controla cuál archivo de recurso utiliza el agente maestro. Este archivo contiene información de la configuración del agente y la ruta predeterminada; el nombre del archivo es `/etc/snmp/conf/snmpdx.rsrc`. Debido a que el agente maestro es arrancado en el momento de inicio del sistema, estas opciones deben cambiarse en el archivo de inicio del sistema `/etc/init.d/init.snmpdx`, para que éstas puedan ser permanentes. Sin embargo, se recomienda que cualquier cambio al archivo de inicio se pruebe antes de que éste sea ejecutado al reinicio del sistema.

¿Por qué cambiar la localización de los archivos? Bueno, en algunos lugares, la práctica estándar es utilizar una localización especial del sistema de archivos, que se ha configurado previamente, y está por lo general disponible a través de todos los sistemas en la red. Además, esta área puede archivarse más regularmente que otros sistemas de archivo y, como resultado, utilizarse para almacenar todos los archivos de configuración de herramientas de terceros. Por esta razón, usar esta área puede ser más beneficioso que usar la localización predeterminada. Igualmente, si se quieren probar nuevas configuraciones que puedan existir en otras localizaciones y con diferentes nombres de archivos, estas opciones pueden ayudar también.

La opción `-n` es útil porque muestra el resumen de la ayuda del agente maestro. Ejecutar el comando `snmpdx` del agente maestro con la opción `-h` mostrará las opciones de línea de comandos disponibles y sus argumentos asociados:

```
# /usr/lib/snmp/snmpdx -h
Jsage: snmpdx [-h]
        [-h]
        [-n relay_agent_name (default relay agent)]
        [-p port (default 161)]
        [-r resource-file (default /etc/snmp/conf/snmpdx.rsrc)]
        [-a access-control-file (default /etc/snmp/conf/snmpdx.acl)]
        [ c config-dir (default /etc/snmp/conf)]
        [-i pid-file (default /var/snmp/snmpdx.st)]
        [-o enterprise-oid-file (default /etc/snmp/conf/
            enterprises.oid)]
        [ y]
        [-m GROUP | SPLIT (default GROUP)]
        [-d trace-level (range 0..4, default 0)]
```

El agente maestro suministra a los mensajes la capacidad de ser mostrados por medio de la opción `d`. Esta opción puede ser muy útil cuando se depuran las configuraciones del agente o problemas operacionales. Los cuatro niveles de depuración son soportados en el rango de 0 (ninguna depuración) a 4 (depuración total). El cuarto nivel de depuración incluye información de configuración del agente, decodificación de cada PDU del mensaje SNMP, información de sesión y contenidos completos de paquete mostrados en hexadecimal. El tercer nivel incluye la configuración del agente, decodificación SNMP e información de sesión, pero no muestra el paquete. El segundo nivel incluye sólo la configuración del agente e información de sesión, y el primer nivel incluye la configuración del agente, pero poca información de sesión.

La opción `m` especifica el modo en el cual se transmiten las solicitudes SNMP para cada subagente. Dos modos están disponibles con el agente maestro. El primero, `GROJP`, indica qué variables múltiples pueden ser incluidas con cada solicitud del agente maestro. El segundo modo, `SPLIT`, asegura que cada variable en la solicitud entrante dé como resultado una solicitud enviada para cada subagente. El modo de transmisión predeterminado es `GROUP`.

La opción `-p` especifica un puerto alternativo que el agente maestro utilizará cuando reciba o transmita mensajes SNMP. El agente maestro utiliza el puerto predefinido SNMP `161` para enviar y recibir mensajes SNMP. Usando esta opción, el agente

maestro escuchará otro puerto especificado después del argumento `-p`. Por ejemplo, el agente maestro escuchará el puerto 200 si se ejecuta el comando siguiente:

```
# /usr/lib/snmp/snmpdx -p 200 -y
```

Recuérdese que si el puerto predeterminado SNMP se cambia, cualquier software de administración SNMP que consulte el agente debe también ser configurado para usar este nuevo puerto. De otra manera, las consultas SNMP realizadas en el puerto antiguo nunca alcanzarán al agente, por tanto, expirarán. Una razón principal para utilizar esta opción es soportar más de una instancia del agente maestro en el mismo sistema. Esta configuración puede utilizarse durante la prueba de una versión más nueva del agente maestro, sin afectar la operación de la versión actual.

La opción `-y` se utiliza cuando el agente maestro se ha caído o reiniciado mientras que uno o más subagentes ya están siendo ejecutados. Esta opción se usa cuando el agente maestro debe recuperar su estado previo antes de la caída, para determinar la configuración y el estado de cada uno de los subagentes definidos. Además, se notará que la opción `-y` se incluye cuando el comando `snmpdx /etc/init.d/init.snmpdx` (localizado en el archivo de inicio del sistema) se ejecuta. Esta secuencia de comandos de inicio ejecuta al agente maestro, como se muestra enseguida:

```
/usr/lib/snmp/snmpdx -y -c /etc/snmp/conf
```

Cuando el agente es iniciado automáticamente por el sistema, el agente maestro no tiene forma de determinar su estado previo y, por tanto, utiliza la opción de recuperación predeterminada. Cualquier subagente que no esté corriendo cuando el agente maestro inicia es reiniciado automáticamente por el agente maestro.

Agente maestro MIB

El agente `snmpdx` implementa una colección de objetos MIB que proporcionan información acerca del estado operacional y configuración del agente maestro y cualquiera de los subagentes configurados. Estos objetos son definidos en el archivo `/var/snmp/mib/snmpdx.mib` bajo el número empresarial de Sun (42) y son referenciados desde el grupo `products` (`enterprises.sun.products`) bajo la rama `sunMasterAgent`. La figura 7.7 muestra la jerarquía del grupo `sunMasterAgent` desde la rama estándar `enterprises`.

Debido al gran número de objetos relacionados con el agente maestro, sólo se describen y listan aquí los objetos más importantes. Aquellos objetos que no se presentan pueden consultarse en el archivo `/var/snmp/mib/snmpdx.mib`, para información adicional.

El grupo `sunMasterAgent` incluye la siguiente colección de objetos MIB:

- objetos globales del agente
- objetos de configuración del subagente

Objetos globales del agente

Los objetos globales del agente proporcionan información de configuración para el agente maestro y todos los subagentes registrados. Esta información es importante porque puede llegar a ser necesaria para sondear al agente maestro directamente, para ayudar a depurar tanto los asuntos operacionales del agente maestro y de los subagentes como otros problemas. Los objetos se listan enseguida:

Nombre del objeto:	`sunMasterAgentStatusFile`
OID:	`sunMasterAgent.1`
Tipo de objeto:	`Display String`
Modo de acceso:	`lectura y escritura`
Descripción:	Este archivo contiene una lista de todas las identificaciones de los procesos del subagente y lo utiliza el agente maestro para recuperación, ante una detención inesperada

Nombre del objeto:	`sunMasterAgentResourceConfigFile`
OID:	`sunMasterAgent.2`
Tipo de objeto:	`Display String`
Modo de acceso:	`lectura y escritura`
Descripción:	Este archivo se utiliza para guardar información de la configuración del subagente y lo utiliza el agente maestro en el inicio

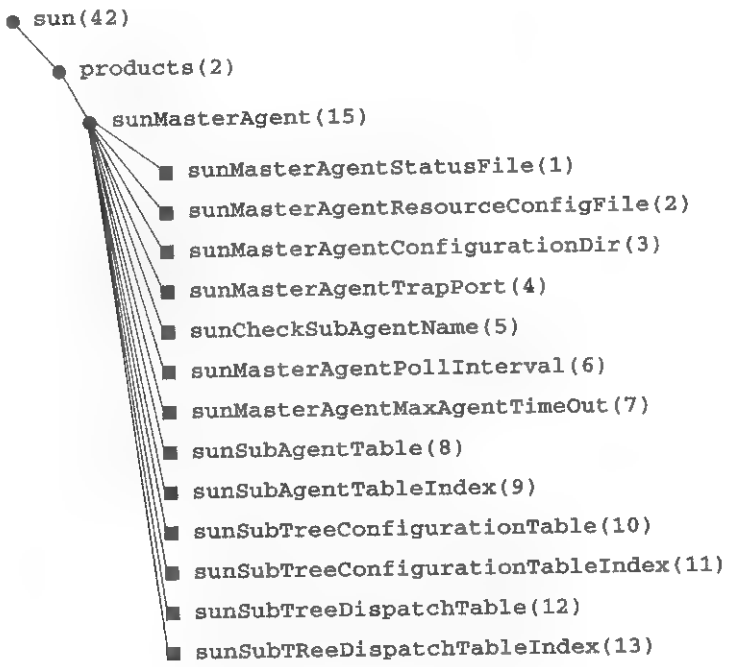


Figura 7.7 Vista jerárquica de `sunMasterAgent`.

Nombre del objeto:	`sunMasterAgentConfigurationDir`
OID:	`sunMasterAgent.3`
Tipo de objeto:	`Display String`
Modo de acceso:	`lectura y escritura`
Descripción:	Es el directorio que contiene los archivos de configuración para el agente maestro

Nombre del objeto:	`sunMasterAgentTrapPort`
OID:	`sunMasterAgent.3`
Tipo de objeto:	`Integer`
Modo de acceso:	`sólo lectura`

Descripción: Es el puerto utilizado por el agente maestro para recibir trampas SNMP desde cada uno de los subagentes. Estas trampas son transmitidas luego a uno o más administradores encontrados en el archivo de configuración del agente maestro

Nombre del objeto: `sunCheckSubAgentName`
OID: `sunMasterAgent.5`
Tipo de objeto: `Display String`
Modo de acceso: `lectura y escritura`
Descripción: Los subagentes lo utilizan para verificar que ningún subagente duplicado con el mismo nombre está corriendo

Nombre del objeto: `sunMasterAgentPollInterval`
OID: `sunMasterAgent.6`
Tipo de objeto: `Integer`
Modo de acceso: `sólo lectura`
Descripción: Esta variable contiene el intervalo de tiempo en el que el agente ejecutará actividades de limpieza, como determinar si el archivo de recurso ha cambiado, redescubrir cada subagente, y otras tareas relacionadas

Nombre del objeto: `sunMasterAgentMaxAgentTimeOut`
OID: `sunMasterAgent.7`
Tipo de objeto: `Integer`
Modo de acceso: `sólo lectura`

Descripción: Este objeto se utiliza para señalar la extemporaneidad máxima permitida que un subagente puede establecer. Lo utiliza el agente maestro para esperar las solicitudes enviadas a un subagente antes de que se considere que la solicitud ha expirado

Objetos de configuración del subagente

Objetos adicionales en el archivo `snmpdx.mib` proporcionan información sobre cada subagente que opera bajo el agente maestro. Esta información está disponible a través de una tabla, y usando objetos de índice. La tabla (`sunSubAgentTable`) lista todos los subagentes registrados con el agente maestro. Cada entrada en esta tabla se describe como un objeto `sunSubAgentEntry`. El acceso a la tabla `sunSubAgentTable` se logra a través del objeto `sunSubAgentEntry`. La figura 7.8 muestra la jerarquía del grupo `sunSubAgentTable` desde la rama estándar `enterprises`.

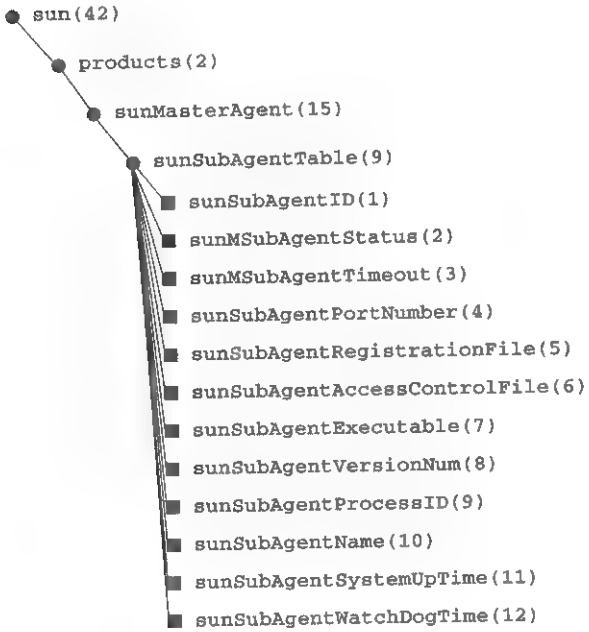


Figura 7.8 Vista de árbol de `sunSubAgentEntry`.

La tabla `sunSubAgentEntry` consta de los siguientes objetos relacionados con el subagente:

Nombre del objeto:	`sunSubAgentID`
OID:	`sunSubAgentEntry.1`
Tipo de objeto:	`Integer`
Modo de acceso:	`sólo lectura`
Descripción:	Ésta es una identificación (ID) única para cada subagente que corre bajo el agente maestro

Nombre del objeto:	`sunSubAgentStatus`
OID:	`sunSubAgentEntry.2`
Tipo de objeto:	`Integer`
Modo de acceso:	`lectura y escritura`
Descripción:	Indica el estado del subagente. Los posibles estados incluyen `init (1)`, `load (2)`, `active (3)`, `inactive (4)` y `destroy (5)`

Nombre del objeto:	`sunSubAgentTimeout`
OID:	`sunSubAgentEntry.3`
Tipo de objeto:	`Integer`
Modo de acceso:	`lectura y escritura`
Descripción:	Cantidad máxima de tiempo que un agente maestro esperará a un subagente para completar una solicitud, expresada en microsegundos

Nombre del objeto:	`sunSubAgentPortNumber`
OID:	`sunSubAgentEntry.4`
Tipo de objeto:	`Integer`
Modo de acceso:	`lectura y escritura`
Descripción:	Número de puerto utilizado por el subagente para escuchar solicitudes del agente maestro

Nombre del objeto:	`sunSubAgentRegistrationFile`
OID:	`sunSubAgentEntry.5`
Tipo de objeto:	`Display String`
Modo de acceso:	`lectura y escritura`
Descripción:	Especifica el archivo de registro utilizado por el subagente. Cada subagente debe tener su propio archivo y éste contiene el nombre del subagente, información del árbol MIB, puerto preferido SNMP, etc.

Nombre del objeto:	`sunSubAgentAccessControlFile`
OID:	`sunSubAgentEntry.6`
Tipo de objeto:	`Display String`
Modo de acceso:	`lectura y escritura`
Descripción:	Especifica el archivo de control de acceso utilizado por el subagente para almacenar información de comunidad SNMP

Nombre del objeto:	`sunSubAgentExecutable`
OID:	`sunSubAgentEntry.7`

Tipo de objeto:	`Display String`
Modo de acceso:	`lectura y escritura`
Descripción:	Contiene el programa ejecutable del subagente

Nombre del objeto:	`sunSubAgentVersionNum`
OID:	`sunSubAgentEntry.8`
Tipo de objeto:	`Display String`
Modo de acceso:	`lectura y escritura`
Descripción:	Información de versión para el subagente

Nombre del objeto:	`sunSubAgentProcessID`
OID:	`sunSubAgentEntry.9`
Tipo de objeto:	`Display String`
Modo de acceso:	`lectura y escritura`
Descripción:	ID del proceso del subagente

Nombre del objeto:	`sunSubAgentName`
OID:	`sunSubAgentEntry.10`
Tipo de objeto:	`Display String`
Modo de acceso:	`lectura y escritura`
Descripción:	Nombre del subagente especificado en el archivo de registro

Nombre del objeto:	`sunSubAgentSystemUpTime`
OID:	`sunSubAgentEntry.11`

Tipo de objeto:	`TimeTicks`
Modo de acceso:	`lectura y escritura`
Descripción:	Tiempo que el subajente ha estado en ejecución

Nombre del objeto:	`sunSubAgentWatchDogTime`
OID:	`sunSubAgentEntry.12`
Tipo de objeto:	`Integer`
Modo de acceso:	`lectura y escritura`
Descripción:	Cantidad de tiempo que el agente maestro espera antes de sondear al subagente para asegurarse de que éste aún opera correctamente, expresado en segundos

El agente SNMP de Sun

El agente Sun proporciona los siguientes servicios:

- obtiene estadísticas de red
- obtiene información de procesos del sistema
- obtiene información de la memoria del sistema
- envía trampas a los sistemas específicos

Esta información de sistema y de red estará disponible sondeando la MIB del agente. El agente soporta tanto a la MIB-II como a porciones de la MIB privada de Sun. La MIB Sun se describe enseguida. La definición de *trampa*, la cual debe hacerse en el archivo de configuración del agente, también se describe a continuación.

Opciones de línea de comandos

La utilidad `mibiisa` soporta varias opciones de línea de comandos, como se resume en la tabla 7.5. Tal vez una de las más importantes es la opción `-r`, la cual puede usarse para inhabilitar el acceso de escritura a la MIB del agente. Debido a que la utilidad

`mibiisa` implementa el SNMPv1, el agente no suministra un modelo de seguridad muy robusto. Puede ser necesario limitar el acceso a los objetos del sistema soportados por el agente. Usando esta opción, todas las `set-request` SNMP serán rechazadas, pero el acceso de lectura operará sin ser afectado.

Cualquier intento para alterar o establecer un objeto MIB después de utilizar la opción `-r`, causará un error. Más específicamente, cuando se usa un explorador MIB, por ejemplo, si se intenta establecer objetos MIB que están colocados en el modo de sólo lectura, se generará el mensaje de trampa de autenticación.

Tabla 7.5 Opciones de comandos del agente maestro.

Opción	Descripción
`-a`	Inhabilita las trampas de autenticación
`-c`	Especifica la ruta para buscar el archivo de configuración
`-d`	Habilita las salidas de depuración
`-p`	Utiliza un puerto especificado, en lugar del `161`
`-r`	Coloca la MIB en el modo de sólo lectura

El directorio en el cual el `mibiisa` busca el archivo de configuración requerido está predeterminado como `/etc/snmp/conf`. Para especificar un directorio alterno, se utiliza la opción `-c` seguida del nombre del directorio. Esta opción puede utilizarse para ayudar a probar nuevos archivos de configuración que pueden residir en otras ubicaciones del directorio. El agente `mibiisa` requiere un archivo de configuración, y si éste no puede localizar una configuración válida, saldrá con un error.

Cuando una entidad SNMP intenta sondear un agente con la cadena de comunidad incorrecta, se despacha un mensaje de autenticación hacia la dirección configurada en el archivo de configuración `mibiisa`. Si se quisiera inhabilitar estos mensajes de trampa, se utiliza la opción `-a`.

Para colocar al agente en el modo depurar, puede utilizarse la opción `-d` cuando se invoca al agente en la línea de comandos.

Además, puede incluirse un nivel opcional de depuración que disminuya los niveles de depuración desde 3 (el más alto) hasta 1 (el más bajo).

Dependiendo de los servicios de trabajo en red que están corriendo actualmente en el sistema, puede ser necesario usar un puerto UDP diferente, en lugar del predeterminado 161. Esto se requerirá si el agente `mibiisa` se ejecuta con un agente maestro. En este caso, el agente necesitará ser arrancado en un puerto diferente, usando la opción `-p`. Por ejemplo, para arrancar el agente en el puerto 32000, se utiliza el siguiente comando:

```
# /usr/lib/snmp/mibiisa -p 32000
```

Configuración del agente Sun

El agente `mibiisa` utiliza un archivo de configuración llamado `snmpd.conf` y, de manera predeterminada, busca el directorio `/etc/snmp/conf` para este archivo. Este archivo contiene información de configuración que controla el acceso a la MIB del agente e información adicional de control. Cada entrada contiene una palabra clave seguida de un valor de cadena, separado por uno o más espacios en blanco. Los comentarios también son soportados en este archivo mediante el uso del signo número (#) y son ignorados cuando son procesados por el agente. Las palabras clave que pueden utilizarse se resumen y describen con más detalle en la tabla 7.6.

Tabla 7.6 Opciones de configuración del agente.

Palabra clave	Descripción
sysdescr	Información del sistema del agente
syscontact	Información de contacto del sistema
syslocation	Información sobre la localización física del sistema
trap	Envía trampas a *hosts* específicos
system-group-read-community	Controla el acceso de lectura a la MIB-II y a los objetos del sistema de Sun

Tabla 7.6 Opciones de configuración del agente. (*continuación*)

Palabra clave	Descripción
system-group-write-community	Controla el acceso de escritura a la MIB-II y a los objetos del sistema de Sun
read-community	Controla el acceso de lectura a toda la MIB
write-community	Controla el acceso de escritura a toda la MIB
trap-community	Cadena de trampa de comunidad
kernel-file	Archivo de símbolos de búsqueda
managers	*Hosts* a los que se les permite comunicarse con el agente

La palabra clave **sysdescr** se refiere al objeto MIB-II `sysDescr`, y el valor asociado lo utiliza el agente cuando responde a solicitudes de este objeto. Ésta describe al sistema en el cual el agente se ejecuta (por ejemplo, `Ultra SPARC 2` o `SPARCclassic`). Los valores para este objeto pueden incluir cualquier cadena, pero deberían representar alguna información relacionada con el tipo de sistema y, tal vez, también información del sistema operativo.

La palabra clave `syscontact` se refiere al objeto MIB-II `sysContact`, y el valor asociado lo utiliza el agente cuando responde a solicitudes de este objeto. El objeto `syscontact` se utiliza para describir el grupo de contacto o nombre que es responsable por el sistema (por ejemplo, `Steve Rossman x3414`).

La palabra clave **syslocation** se refiere al objeto MIB-II `sysLocation`, y el valor asociado lo utiliza el agente cuando responde a solicitudes de este objeto. El objeto `sysLocation` se utiliza para describir la localización física del sistema (por ejemplo, `Building 2, Graphics Lab 11-B`).

La palabra clave **trap** especifica uno o más *hosts* que deberían recibir trampas desde el agente. Hasta cinco *hosts* pueden incluirse con esta opción. Puede utilizarse la dirección IP o los nombres del *host*. Cualquiera de los *hosts* adicionales más allá del máximo será ignorado. Uno de los usos para esta palabra clave es transmitir trampas de autenticación cuando una entidad

SNMP intenta sondear al agente con una cadena de comunidad incorrecta. Ésta es la manera principal de determinar si otro grupo está intentando tener acceso a un dispositivo sin tener la autorización apropiada.

Una trampa de autorización puede indicar también que se cambió la cadena de comunidad en el lado del agente, y este cambio no fue propagado también hacia el lado del administrador. También puede significar que se instaló recientemente un paquete nuevo de software de administración SNMP que utiliza las cadenas de comunidad de `public` o `private`, y que estas cadenas deben ajustarse a las cadenas utilizadas en el sitio local.

Las palabras clave **system-group-read-community** y **system-group-write-community** controlan el acceso de lectura-escritura al agente MIB que incluye sólo al grupo de sistema en la MIB-II y a los grupos de objetos de Sun. Las palabras clave **read-community** y **write-community** controlan el acceso de lectura-escritura para toda la MIB. La palabra clave **trap-community** especifica la cadena de comunidad para mensajes de trampa. La palabra clave **kernel-file** especifica dónde el agente debería buscar símbolos.

La palabra clave **managers** controla a cuáles *hosts* se les permite comunicarse con el agente, ya sea por el acceso de escritura o por el de lectura. Ésta forma una lista básica de control de acceso (ACL) para el agente. La ACL es una muy buena forma de abordar algunas fallas de seguridad dentro del modelo de seguridad SNMP. Sin embargo, téngase en cuenta que esta opción proporciona un mecanismo fundamental de autenticación, que es vulnerable dada la situación correcta. Por ejemplo, la ACL puede ser engañada si el sistema `chips` estaba en la lista de acceso, pero el sistema `dip` supuso la identificación de `chip` mientras éste estaba abajo (*down*) por reparaciones. Por tanto, la opción **managers** debe considerarse sólo como una parte de la estrategia global de seguridad.

Ejemplo de un archivo de configuración del agente

El listado 7.7 ilustra aún más las opciones del archivo de configuración del agente.

Listado 7.7 Ejemplo de un archivo de configuración del agente.

```
#
# System Information
#
sysdescr          SPARC Ultra 60
syscontact        Steve Rossman
syslocation       Science Center room 22B
#
# Community Strings (system group only)
#
system group-read-community      louvrehasart
system-group-write community     louvrehasart
#
# Community Strings (entire MIB)
#
read-community   louvrenasart_1834
write community  louvrenasart_1824
#
# Trap Information
#
trap              rembrandt
trap-community    louvrehasart
#
# Standard Unix kernel
#
kernel-file       /vmunix
#
# Which systems are permitted to talk to this agent
#
managers          rembrandt monet
```

La información del sistema (`sysdescr`, `syscontact` y `syslocation`) debería especificarse para cada sistema en el cual el agente está corriendo. En este ejemplo, el agente está corriendo en una estación de trabajo `SPARC Ultra 60`, el contacto para el sistema es `Steve Rossman`, y el sistema está localizado en el `Science Center room 22B`. Nótese que puede incluirse cualquier información descriptiva. Por tanto, el `sysdescr` no puede representar el tipo real de sistema en el que se ejecuta el agente. A los objetos MIB del sistema se tiene acceso con la cadena de comunidad de lectura y escritura `louvrehasart`. El acceso al resto de la MIB tanto de lectura como de escritura está establecido a la cadena `louvrehasart_1824`. La cadena de comunidad del sistema es diferente de toda la MIB, en el caso que se quiera que otros tengan acceso a la información del sistema (con propósitos de inventario, por ejemplo) sin tener acceso al núcleo de otros objetos del agente. Todas las trampas generadas desde este

agente serán enviadas al *host* `rembrandt` con la cadena de comunidad de `louvrenasart`. Usando la palabra clave **managers** como la ACL básica, al agente se le permite tener acceso a solicitudes SNMP desde los *hosts* `rembrandt` y `monet`, y todas las otras solicitudes serán descartadas por el agente.

Objetos MIB del agente Sun

El agente `mibiisa` soporta todos los objetos definidos en la MIB-II, más objetos adicionales bajo el número de vendedor amplio Sun enterprises (`42`). Los objetos MIB-II se describieron en el capítulo 3, "Conjunto de protocolos TCP/IP". Sin embargo, Sun ha cambiado algunos de los atributos relacionados con los valores de retorno o con los modos de acceso de las definiciones estándar MIB-II. Esta sección describe estos cambios y los objetos Sun adicionales.

Diferencias de la MIB Sun

Las diferencias entre los objetos `mibiisa` y los objetos MIB-II estándar incluyen las siguientes:

- El acceso a los objetos MIB ha cambiado
- Ciertos objetos MIB contienen valores fijos

La tabla 7.7 resume estas diferencias y contiene una lista de objetos MIB, banderas que indican las diferencias de los objetos, el método de acceso (ya sea *lectura-escritura* o *sólo lectura*), y el valor de retorno, si hay alguno. Las banderas de objeto `V` y `A` se refieren a si el cambio del objeto está relacionado con el valor (`V`) del objeto o con el modo de acceso (`A`) del objeto. En la columna se coloca una marca de verificación para indicar cuál atributo del objeto es diferente de las definiciones estándar MIB-II. Por ejemplo, el modo de acceso del objeto `sysName` es de lectura-escritura en la MIB-II, mientras que la MIB Sun proporciona acceso de sólo lectura. Como resultado, en la columna se coloca una marca de verificación. La mayoría de los objetos Sun, en esta tabla, sólo retornan un valor fijo. En muchos de estos casos, se retorna un `1`, lo que significa que el agente no soporta el objeto y éste es proporcionado sólo para propósitos de compatibilidad.

Tabla 7.7 Comparación entre los objetos MIB-II y los objetos Sun MIB.

Objeto MIB	V	A	Acceso en MIB-II	Acceso en mibiisa	Valor en mibiisa
sysName		✔	lectura-escritura	sólo lectura	
atIfIndex		✔	lectura-escritura	sólo lectura	
ipDefaultTTL		✔	lectura-escritura	sólo lectura	
ipRoutIfIndex	✔		lectura-escritura	lectura-escritura	(fijo)
ipRouteMetric1	✔		lectura-escritura	lectura-escritura	-1
ipRouteMetric2	✔		lectura-escritura	lectura-escritura	-1
ipRouteMetric3	✔		lectura-escritura	lectura-escritura	-1
ipRouteMetric4	✔		lectura-escritura	lectura-escritura	-1
ipRouteMetric5	✔		lectura-escritura	lectura-escritura	-1
ipRouteType	✔		lectura-escritura	lectura-escritura	(fijo)
ipRouteAge	✔		lectura-escritura	lectura-escritura	0
ipRouteMask	✔		lectura-escritura	lectura-escritura	(fijo)
icmpInDestUnreachs	✔		sólo lectura	sólo lectura	1
icmpInTimeExcds	✔		sólo lectura	sólo lectura	1

Tabla 7.7 Comparación entre los objetos MIB-II y los objetos Sun MIB. (*continuación*)

Objeto MIB	V	A	Acceso en MIB-II	Acceso en mibiisa	Valor en mibiisa
icmpInParmProbs	✔		sólo lectura	sólo lectura	1
icmpInSrcQuenchs	✔		sólo lectura	sólo lectura	1
icmpInRedirects	✔		sólo lectura	sólo lectura	1
icmpInEchos	✔		sólo lectura	sólo lectura	1
icmpInEchoReps	✔		sólo lectura	sólo lectura	1
icmpInTimestamps	✔		sólo lectura	sólo lectura	1
icmpInTimestamp-Reps	✔		sólo lectura	sólo lectura	1
icmpInAddrMasks	✔		sólo lectura	sólo lectura	1
icmpInAddrMaskReps	✔		sólo lectura	sólo lectura	1
icmpOutDestUn-reachs	✔		sólo lectura	sólo lectura	1
icmpOutTimeExcds	✔		sólo lectura	sólo lectura	1
icmpOutParmProbs	✔		sólo lectura	sólo lectura	1
icmpOutSrcQuenchs	✔		sólo lectura	sólo lectura	1
icmpOutRedirects	✔		sólo lectura	sólo lectura	1
icmpOutEchos	✔		sólo lectura	sólo lectura	1
icmpOutEchoReps	✔		sólo lectura	sólo lectura	1
icmpOutTimestamps	✔		sólo lectura	sólo lectura	1
icmpOutTimestamp-Reps	✔		sólo lectura	sólo lectura	1
icmpOutAddrMasks	✔		sólo lectura	sólo lectura	1
icmpOutAddrMask-Reps	✔		sólo lectura	sólo lectura	1
ifInUnknownProtos	✔		sólo lectura	sólo lectura	1

Tabla 7.7 Comparación entre los objetos MIB-II y los objetos Sun MIB. (*continuación*)

Objeto MIB	V	A	Acceso en MIB-II	Acceso en mibiisa	Valor en mibiisa
ipAdEntBcastAddr	✔		sólo lectura	sólo lectura	1
ipAdEntReasmMaxSiz	✔		sólo lectura	sólo lectura	65535
ipNetToMediaType	✔		sólo lectura	sólo lectura	Dinámica de retorno (3)
ipRoutingDiscards	✔		sólo lectura	sólo lectura	0

Objetos MIB Sun Enterprise

Los objetos Sun enterprise soportados por `mibiisa` incluyen los siguientes grupos:

- `sunSystem`
- `sunProcessTable`
- `sunHostPerf`

El grupo `sunSystem` suministra información genérica del *host*, como la información del sistema y del agente. El grupo `sunProcessTable` contiene información respecto a los procesos del sistema que actualmente están siendo ejecutados, que se encuentran en el sistema del agente. El grupo `sonHostPerf` contiene información relacionada con el rendimiento. Cada uno de estos grupos se describe completamente enseguida. Estos grupos hacen parte de la MIB Sun enterprises, la cual contiene grupos adicionales. Sin embargo, el agente sólo soporta los tres grupos listados arriba; la figura 7.9 muestra la representación jerárquica de todos los grupos bajo la MIB Sun de la rama `enterprises`.

El grupo `sunSystem` consta de objetos que pueden usarse para obtener información del sistema, como la descripción del agente del sistema, la cadena de identificación del *host*, parte del archivo del mensaje del día, y el tiempo del sistema. La figura 7.10 muestra la representación jerárquica de los objetos encontrados en este grupo a partir de la rama `sun`.

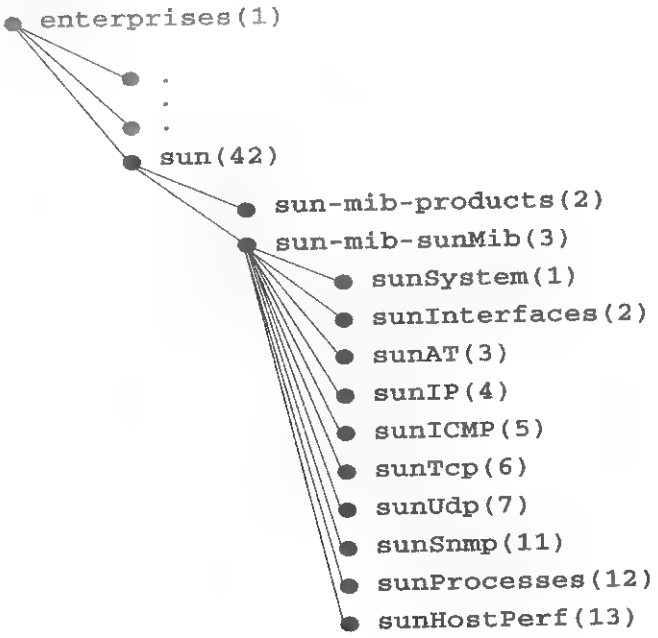


Figura 7.9 MIB Sun enterprise.

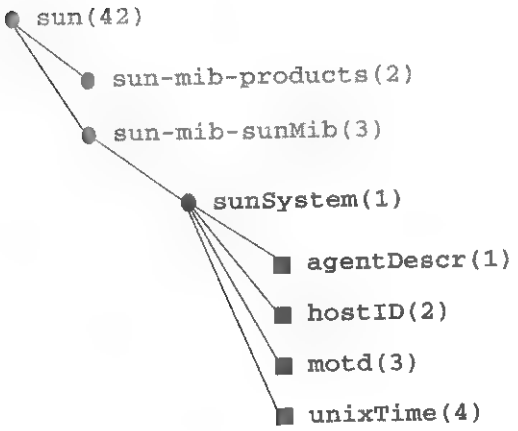


Figura 7.10 Vista de árbol de `sunSystem`.

A continuación aparece una descripción detallada de estos objetos MIB.

Nombre del objeto: `agentDescr`

OID: `sunSystem.1`

Tipo de objeto: `Display String [255]`

Modo de acceso: `sólo lectura`

Descripción: Es una descripción del agente SNMP, que se usa para identificarse a sí mismo con otras entidades, como el administrador SNMP. El agente asigna la cadena de valor `Sun Microsystems SNMP Agent`

Nombre del objeto: `hostID`

OID: `sunSystem.2`

Tipo de objeto: `Display String [255]`

Modo de acceso: `sólo lectura`

Descripción: Identificador de hardware de cuatro bytes, el cual debe ser único para cada sistema Sun. Este valor equivale a la ejecución de la utilidad `hostid` en la línea de comandos UNIX

Nombre del objeto: `motd`

OID: `sunSystem.3`

Tipo de objeto: `Display String [4]`

Modo de acceso: `sólo lectura`

Descripción:	Primera línea del archivo `/etc/motd`. El archivo `/etc/motd` es el archivo de mensaje del día, en el cual se presentarán los contenidos del archivo cada vez que un usuario entre en el sistema. De manera predeterminada, la primera línea contendrá la versión del sistema operativo e información relacionada, pero puede contener cualquier otra información que el administrador quiera poner a disposición
Nombre del objeto:	`unixTime`
OID:	`sunSystem.4`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Es el tiempo de reloj del sistema UNIX, medido en segundos desde 1/1/1970

Grupo de tabla de proceso Sun

La tabla `sunProcessTable` consta de objetos que pueden usarse para obtener una lista de los procesos del sistema e información relacionada. La figura 7.11 muestra una representación jerárquica de estos objetos y cómo están relacionados con la rama principal `sun`. El objeto `psEntry` es el índice para la tabla `sunProcessTable` y se utiliza para tener acceso a cada miembro, el cual representa un proceso UNIX diferente. Debido a que cada objeto listado enseguida es un miembro de la `sunProcessTable`, la OID se representa como el índice desde el punto de vista del objeto `psEntry`.

La información disponible en este grupo también puede obtenerse y compararse con el comando **ps** del sistema operativo. Para entender mejor cómo se utiliza este grupo de objetos, la figura 7.12 muestra los objetos que corresponden a los campos del comando **ps**.

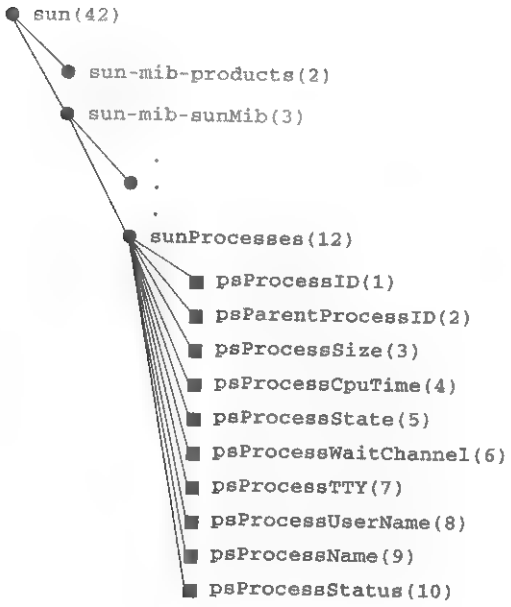


Figura 7.11 Vista de la `sunProcessTable`.

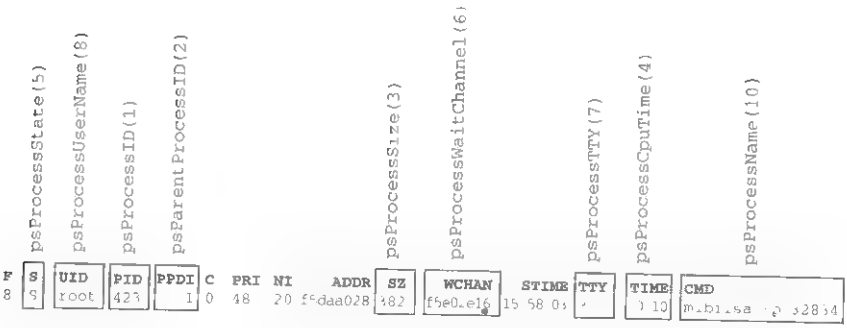


Figura 7.12 Transformación del estado del proceso MIB.

La tabla de proceso contiene los siguientes objetos:

Nombre del objeto: `psProcessID`

OID: `psEntry.1`

Tipo de objeto: `Counter`

Modo de acceso:	sólo lectura
Descripción:	Presenta la identificación del proceso

Nombre del objeto:	psParentProcessID
OID:	psEntry.2
Tipo de objeto:	Counter
Modo de acceso:	sólo lectura
Descripción:	Presenta la identificación del proceso padre

Nombre del objeto:	psProcessSize
OID:	psEntry.3
Tipo de objeto:	Counter
Modo de acceso:	sólo lectura
Descripción:	Presenta el tamaño de la memoria que está usando el proceso, incluidos los segmentos de datos y de pila

Nombre del objeto:	psProcessCpuTime
OID:	psEntry.4
Tipo de objeto:	Counter
Modo de acceso:	sólo lectura
Descripción:	Presenta la cantidad de tiempo de la unidad central de procesamiento (CPU), e incluye tanto el tiempo del usuario como el del sistema que este proceso ha consumido en el instante que este valor es recuperado

Nombre del objeto:	psProcessState
OID:	psEntry.5
Tipo de objeto:	Display String [4]
Modo de acceso:	sólo lectura
Descripción:	Presenta el estado de ejecución actual del proceso. Esto incluye el uso de las siguientes banderas: R: corriendo T: detenido P: en espera de página D: en espera no interrumpible S: durmiendo (menos de 20 segundos) I: inactivo (más de 20 segundos) Z: zombi

Nombre del objeto:	psProcessWaitChannel
OID:	psEntry.6
Tipo de objeto:	Display String [16]
Modo de acceso:	sólo lectura
Descripción:	Presenta la razón por la cual el proceso está en modo de espera

Nombre del objeto:	psProcessTTY
OID:	psEntry.7
Tipo de objeto:	Display String [16]
Modo de acceso:	sólo lectura
Descripción:	Presenta la terminal de control de este proceso, si hay alguna

Nombre del objeto:	psProcessUserName
OID:	psEntry.8
Tipo de objeto:	Display String [16]
Modo de acceso:	sólo lectura
Descripción:	Presenta el nombre del usuario asociado con este proceso

Nombre del objeto:	psProcessUserID
OID:	psEntry.9
Tipo de objeto:	Counter
Modo de acceso:	sólo lectura
Descripción:	Presenta la identificación numérica del usuario asociada con la cadena psProcessUserName

Nombre del objeto:	psProcessName
OID:	psEntry.10
Tipo de objeto:	Display String [64]
Modo de acceso:	sólo lectura
Descripción:	Presenta el nombre de la cadena de comandos para este proceso

Nombre del objeto:	psProcessStatus
OID:	psEntry.11
Tipo de objeto:	Counter
Modo de acceso:	lectura y escritura
Descripción:	Indica la señal para enviar este proceso

Grupo de rendimiento Sun Host

El grupo `sunHostPerf` consta de una colección de objetos relacionados con el rendimiento del sistema, que comprende desde el número total de *timeticks* utilizados por un proceso hasta el número total de colisiones de red. Debido a la cantidad tan grande de estos objetos, solamente se describirán los más útiles. Si se desea conocer la lista completa de objetos disponibles en este grupo, consúltese el archivo MIB Sun `/var/snmp/mib/sun.mib`, para información adicional.

Entre los objetos más útiles en el grupo `sunHostPerf` están los objetos relacionados con el sistema de intercambio, interrupciones de dispositivos y tráfico de red. La figura 7.13 muestra una representación jerárquica de estos objetos y cómo se relacionan con la rama principal `sun`. Los objetos de tráfico son contadores, los cuales representan totales para todo el sistema, sin tener en cuenta el número de conexiones a la red, y son similares a la salida del comando **netstat**. Por ejemplo, el objeto `rsIfInPackets` suministra el número total de paquetes de entrada desde la red para todas las interfaces de red definidas en el sistema.

El grupo `sunHostPerf` incluye los siguientes objetos:

Nombre del objeto:	`rsVpagesIn`
OID:	`sunHostPerf.9`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número total de páginas leídas desde el disco a partir de la última reiniciación

Nombre del objeto:	`rsVpagesOut`
OID:	`sunHostPerf.9`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número total de páginas escritas al disco desde la última reiniciación

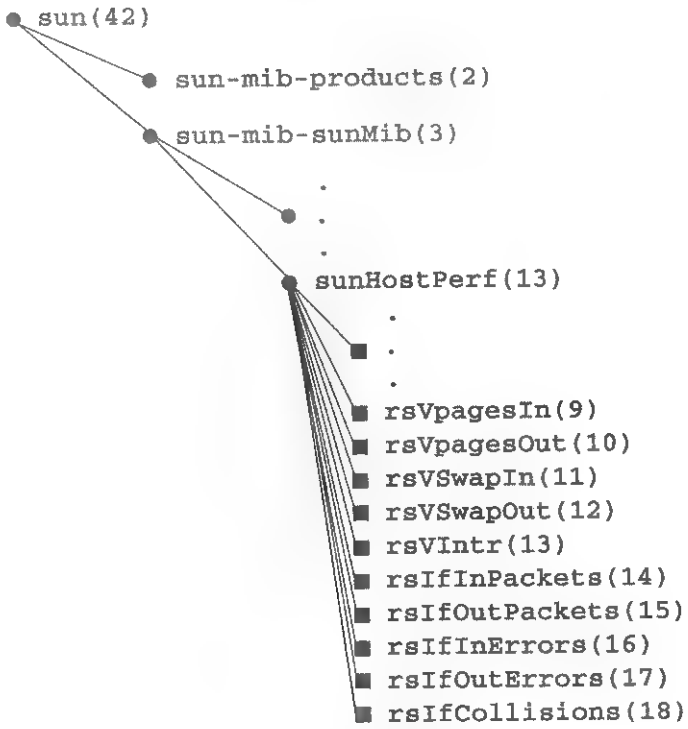


Figura 7.13 Vista de `sunHostPerf`.

Nombre del objeto:	`rsVSwapIn`
OID:	`sunHostPerf.11`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número total de páginas intercambiadas a partir de la última reiniciación

Nombre del objeto:	`rsVSwapOut`
OID:	`sunHostPerf.12`
Tipo de objeto:	`Counter`

Modo de acceso:	`sólo lectura`
Descripción:	Número total de páginas leídas del disco, a partir de la última reiniciación

Nombre del objeto:	`rsVIntr`
OID:	`sunHostPerf.13`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número total de interrupciones de dispositivos a partir de la última reiniciación; incluye las interrupciones para todos los dispositivos, como discos y otros dispositivos periféricos

Nombre del objeto:	`rsIfInPackets`
OID:	`sunHostPerf.14`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número total de paquetes de entrada obtenidos de la red a partir de la última reiniciación

Nombre del objeto:	`rsIfInPackets`
OID:	`sunHostPerf.15`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número total de paquetes de salida transmitidos a la red, a partir de la última reiniciación

Nombre del objeto:	`rsIfInErrors`
OID:	`sunHostPerf.16`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número total de errores de entrada de la red a partir de la última reiniciación

Nombre del objeto:	`rsIfOutErrors`
OID:	`sunHostPerf.17`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número total de errores de salida encontrados mientras se intenta transmitir a la red, a partir de la última reiniciación

Nombre del objeto:	`rsIfCollisions`
OID:	`sunHostPerf.18`
Tipo de objeto:	`Counter`
Modo de acceso:	`sólo lectura`
Descripción:	Número total de colisiones de salida registradas en la red a partir de la última reiniciación

08

Herramientas SNMP

El capítulo 4, "Herramientas y utilidades centrales del sistema", y el capítulo 5, "Utilidades y herramientas adicionales del sistema", analizan las utilidades y herramientas del sistema operativo. Este capítulo se enfoca más en las herramientas de la red que pueden ayudar a administrar tanto los sistemas como los elementos del trabajo en red, como los conmutadores, los enrutadores, las sondas, los concentradores y otros dispositivos. Todos ellos se basan en el SNMP y pueden utilizarse para interrogar a cualquier agente que dé soporte a SNMP para enriquecer la información.

Este capítulo cubre las siguientes herramientas:

Comando	Página
snmpdelta	429
snmpget	431
snmpgetnext	433
snmpnetstat	435
snmpset	440
snmpstatus	444
snmptable	447
snmptest	450
snmptranslate	450
snmptrap	452
snmptrapd	453
snmpwalk	457
snmpbulkwalk	459
snmpconf	462
tkmib	466

Funciones de seguimiento y administración

Las herramientas analizadas en esta sección suministran acceso a una variedad de información del agente y son particularmente útiles en el seguimiento y la administración. Alguna de la información más importante que puede recolectacse de los dispositivos incluye:

- latido del sistema SNMP
- mensajes del sistema subir/bajar
- estadísticas del protocolo
- mediciones de rendimiento de la interfaz

- actividad de procesos del sistema
- enrutado
- estadísticas de rendimiento

Algunas de las herramientas también pueden utilizarse para configurar los dispositivos de la red. Las secciones que siguen describen cada uno de estos usos.

Latido del sistema

Un latido del sistema es una `get request` SNMP que un administrador utiliza para determinar la accesibilidad general del agente y del sistema. Por ejemplo, el administrador de red puede sondear la variable MIB del reloj del sistema del agente para determinar que cada sondeo sucesivo es más reciente que el anterior. Cada sondeo sucesivo debe indicar que el tiempo está avanzando. La variable MIB que el administrador de red puede sondear es el objeto `unixTime`, parte del grupo `sunSystem` del agente del sistema Sun. Como una alternativa, el administrador de red puede sondear el objeto `sysUpTime` del grupo `system` MIB-2, el cual estaría disponible desde todos los agentes SNMP.

Mensajes del sistema subir/bajar

Si el sistema se baja y reinicia por cualquier razón, se enviará un mensaje hacia el sistema de administración de red designado, en la forma de una trampa (`trap`) SNMP. Recuérdese que una trampa es un mensaje no solicitado emitido desde el agente que indica alguna condición o evento especial. Al recibir estos mensajes, el administrador es informado, por ejemplo, de las caídas del sistema y puede tomar la acción apropiada. El archivo de configuración del agente UCD y las herramientas adicionales pueden utilizarse para transmitir mensajes de trampas a uno o más sistemas de administración de la red.

Estadísticas del protocolo

Debido a que muchos agentes SNMP soportan el estándar MIB-II, son posibles el seguimiento del rendimiento del protocolo y el seguimiento del sistema. Esto incluye el seguimiento del IP, ICMP, TCP y SNMP de los contadores de interfaz de red y objetos adicionales de rendimiento del agente del sistema.

Actividad de procesos del sistema

Con las extensiones MIB Sun y el agente UCD, también es posible el seguimiento de los procesos del sistema. Los objetos pueden establecer el estado general de la red o la condición de un sistema individual mediante el seguimiento de los procesos críticos.

Seguimiento de la ruta

El agente de seguimiento de la ruta puede determinar la configuración de enrutado de un sistema y reportar cualquiera de los errores encontrados. Por ejemplo, si una organización ha determinado que cada máquina debe tener una ruta predeterminada, esto puede verificarse mediante el sondeo del agente SNMP en estos dispositivos.

Rendimiento de la interfaz

La MIB-2 también suministra objetos que contienen información de rendimiento para cada una de las interfaces instaladas en un dispositivo SNMP. Es posible recuperar esta información y realizar un seguimiento de rendimiento de interfaces activas.

Control de configuración

Muchos dispositivos de trabajo en red (por ejemplo, enrutadores y conmutadores) deben configurarse antes de ser utilizados de forma efectiva en una red. Además, como los requerimientos de red cambian, también deben cambiar las configuraciones en estos dispositivos. Las herramientas listadas aquí pueden utilizarse para alterar las configuraciones de los dispositivos.

Comandos UCD

Descripción general

El paquete UCD suministra no sólo un agente SNMP robusto y poderoso, sino también una serie de herramientas prácticas que pueden utilizarse para administrar dispositivos SNMP habilitados para el trabajo en red. Estas herramientas soportan el SNMPv1, el SNMPv2 y el SNMPv3. Se puede seleccionar cuál versión se va a utilizar, con una opción de la línea de comandos. Además, estas utilidades pueden emplearse para construir secuencias de comandos u otros programas, a fin de llevar a cabo funciones complejas de administración de la red o tareas personalizadas. Por ejemplo, el comando **snmpget** puede hacer un seguimiento de ciertas interfaces críticas, para determinar si

una o más de ellas llegan a ser inoperables. En tal caso, el personal de soporte apropiado podría ser notificado automáticamente.

La tabla 8.1 lista las herramientas y utilidades que se suministran con el paquete UCD.

Tabla 8.1 Herramientas UCD SNMP.

Herramienta UCD	Descripción
`snmpdelta`	Hacen un seguimiento a los cambios de las variables SNMP
`snmpget`	Obtiene uno o más objetos MIB
`snmpgetnext`	Recorre constantemente un árbol MIB SNMP y obtiene todos los objetos MIB soportados
`snmpnetstat`	Obtiene información de la configuración de la interfaz del agente
`snmpset`	Establece uno o más objetos MIB en un valor especificado
`snmpstatus`	Obtiene información importante de un objeto MIB
`snmptable`	Obtiene una tabla completa SNMP
`snmptest`	Se comunica con una entidad de agente SNMP
`snmptranslate`	Convierte a los objetos MIB en información más significativa
`snmptrap`	Envía mensajes de trampa SNMP a uno o más administradores
`snmptrapd`	Recupera trampas SNMP desde la red
`snmpwalk`	Obtiene un grupo de objetos MIB relacionados
`snmpbulkwalk`	Obtiene un objeto MIB con una solicitud masiva SNMP
`tkmib`	Examina objetos MIB. Esta herramienta se basa en Tcl/Tk

La sintaxis básica de las herramientas UCD es la siguiente:

```
snmpcmd protocol_version [additional_options] hostname
community object [object]
```

La palabra `snmpcmd` es un separador de lugar que representa uno de los comandos UCD listados en la tabla 8.1. Todas las herramientas UCD soportan esta sintaxis básica del comando. La palabra `protocol_version` puede ser `1`, `2c` o `3`. Este identificador de cadena especifica cuál versión SNMP debería usarse cuando se envían consultas al agente. La opción `1` representa el formato estándar SNMPv1. La opción `2c` indica las diferencias entre las unidades de datos de protocolo SNMP soportadas, pero utiliza el mismo enfoque basado en la comunidad, como en SNMPv1. La opción `3` indica el uso del modelo de seguridad SNMPv3.

El separador de lugar `additional_options` representa opciones que controlan tanto los atributos presentados como el comportamiento operacional de las herramientas. Las opciones de línea de comandos más comúnmente usadas se describen en seguida.

La palabra `nostname` puede reemplazarse por el nombre de cualquier *host* en la red que contenga un agente SNMP que se equipare con la información `protocol version` especificada en la línea de comandos. Puede utilizarse una dirección IP válida, expresada en notación punteada, en lugar de un nombre de *host*.

La palabra `object` representa la OID MIB que recuperaría (en el caso de una `get-request` SNMP) o alteraría (en el caso de una `set request` SNMP). Puede expresarse en notación punteada numérica o punteada de nombre. En el caso de una operación `set request` SNMP, se requerirá información adicional del objeto, como se analiza en la sección "Herramienta snmpset". Nótese que pueden especificarse uno o más objetos en la línea de comandos.

Opciones comunes de línea de comandos

Las herramientas UCD comparten argumentos comunes de línea de comandos. Tener un conjunto central de opciones facilita su recordación y uso. Los argumentos soportados por todos los comandos se dividen en dos categorías: opciones operacionales y de presentación. Las opciones operacionales controlan el comportamiento de cada una de las herramientas mientras que las

opciones de presentación controlan cómo se presentan los objetos MIB, los valores asociados, y otra información.

Opciones de presentación

La tabla 8.2 presenta los argumentos de presentación, los cuales controlan algunos aspectos de la salida. Nótese que la tabla no describe totalmente cada una de las ocho opciones disponibles. Por ejemplo, la opción `-h`, la cual presenta una cadena de ayuda de los argumentos de línea de comandos, no se describe, porque es bastante intuitiva.

Tabla 8.2 Opciones comunes de presentación.

Opción	Descripción
`-d`	Vuelca paquetes SNMP
`-D`	Presenta información de depuración
`-h`	Presenta un mensaje de ayuda
`-f`	Presenta una ruta completa del identificador de objetos
`-q`	Facilita el análisis sintáctico de los programas
`-s`	Presenta sólo identificadores de sufijos
`-S`	Presenta tanto identificadores de sufijos como el nombre MIB
`-V`	Presenta información de versión

Tres de estas opciones suministran control sobre cómo se formatea y presenta la información de la ruta MIB: `-f`, `-s` y `-S`. La opción `-f` presenta toda la información de la ruta completa del identificador del objeto. Por tanto, la opción `-f` presentará el objeto:

```
system.sysContact.0
```

como se muestra:

```
.iso.org.dod.internet.mgmt.mib 2.system.sysContact.0
```

La opción `-s` sólo permite imprimir al sufijo de la OID. Se mostrará la última parte simbólica del identificador del objeto MIB. Por ejemplo, la opción `-s` imprimirá:

```
.iso.org.dod.internet.mgmt.mib-2.system.sysName.0
```

como se muestra:

```
SysName.0
```

Finalmente, la opción `-S` solicita que el objeto MIB sea impreso tanto con el sufijo como con el nombre MIB. Por tanto, `-S` presentará:

```
.iso.org.dod.internet.mgmt.mib 2.system.sysUpTime.0
```

como se muestra:

```
SNMPv2-MIB:sysUpTime.0
```

Obsérvese que, en este ejemplo, el objeto `sysJpTime` se encuentra en la MIB SNMPv2; esto es cierto cuando se utilizan las herramientas UCD. Generalmente, sin embargo, este objeto se halla en el árbol MIB-2.

Si se necesita recolectar información SNMP de un dispositivo y usar esta información como entrada en otro programa, la opción `q` será muy útil. Normalmente, la información de un objeto MIB se presenta como se muestra:

```
system.sysObjectID.0 = OID: enterprises.9.1.17
system.sysUpTime.0 = Timeticks: (139494644) 16 days, 3:29:06.44
system.sysContact.0 = Matthew Maxwell
system.sysName.0 = remote-gw5
system.sysLocation.0 = Remote Sales Office (San Jose))
system.sysServices.0 = 6
```

La opción `q`, la cual representa un formato rápido (*quick format*), hace que la salida anterior se formatee de manera diferente. Primero, se renueve el signo igual (=); esto lo hace fácil de analizar sintácticamente porque los datos están ahora en formato de columna. Segundo, nótese que los formatos `sysObjectID` y `sysUpTime` se han alterado. La información de estos dos objetos en el ejemplo precedente se interpreta, mientras, que en el siguiente ejemplo, se presentan sólo los datos elementales:

```
system.sysObjectID.0 enterprises.9.1.17
system.sysUpTime.0 16:3:24:11.44
system.sysContact.0 Matthew Maxwell
system.sysName.0 remote-gw5
system.sysLocation.0 Remote Sales Office (San Jose)
system.sysServices.0 6
```

Para presentar información de depuración, se utiliza la opción `d`. Ésta muestra la información del paquete, incluidos tamaño y destino, y también suministra un vaciado hexadecimal y ASCII del paquete. La salida muestra aquí el resultado de una `get request` SNMP del objeto MIB `system.sysContact`.

```
sending 51 bytes to 10.0.2.220:161:
0000: 30 82 00 2F 02 01 00 04 06 70 75 62 6C 69 63 A0  0../.....public.
0016: 82 00 20 02 04 41 C9 4A 92 02 01 00 02 01 00 30  .. ..A.J.......0
0032: 82 00 10 30 82 00 0C 06 08 2B 06 01 02 01 01 04  ...0.....+......
0048: 00 05 00                                         ...

received 60 bytes from 10.0.2.220:161:
0000: 30 82 00 38 02 01 00 04 06 70 75 62 6C 69 63 A2  0..8 ....publi .
0016: 2B 02 04 41 C9 4A 92 02 01 00 02 01 00 30 1D 30  +..A.J . ... 0 0
0032: 1B 06 08 2B 06 01 02 01 01 04 00 04 0F 4D 61 74  ...+....     .Mat
0048: 74 68 65 77 20 4D 61 78 77 65 6C 6C              thew Maxwell
system.sysContact.0 = Matthew Maxwell
```

La primera parte de la salida es el paquete solicitado, como lo indica la cadena `sending 51 bytes to 10.0.2.220:161`, la cual incluye el formato de paquete SNMP. Nótese la visibilidad de la cadena de comunicación `public`. El paquete recibido es la respuesta del agente, y éste también utiliza el formato del paquete SNMP. En este caso, se ven tanto la cadena de comunidad `public` como la cadena de objeto `sysContact`. La segunda parte es la respuesta, la cual empieza con la cadena `received 60 bytes from 10.0.2.220.161`.

Opciones operacionales

La tabla 8.3 incluye los ocho argumentos operacionales, pero no describe completamente cada uno de ellos. Por ejemplo, debido a que la capacidad de la opción `-c` para definir los valores del reloj con mensajes de autenticación SNMPv2 no es una función crítica para el uso de las herramientas, ésta no se describe.

Tabla 8.3 Opciones operacionales comunes.

Opción	Descripción
`-c`	Establece los valores del reloj
`-m`	Especifica una lista de módulos MIB para cargar
`-M`	Especifica una lista de directorios para buscar por archivos MIB
`-p`	Utiliza el puerto especificado para comunicarse con el agente
`-r`	Especifica el número de reintentos
`-R`	Solicita acceso aleatorio a la tabla MIB del agente
`-t`	Especifica la extemporaneidad entre intentos
`-v`	Especifica la versión de protocolo

Los objetos MIB se localizan en lugares estándar predeterminados y bien conocidos en el árbol MIB. Considérese, por ejemplo, el objeto MIB `system.sysContact.0`, el cual se encuentra normalmente en el siguiente árbol:

```
.iso.org.dod.internet.mgmt.mib-2 tree
```

Las herramientas UCD soportan un concepto de MIB de acceso aleatorio. Utilizando este enfoque, el objeto MIB `system.sysContact` puede ingresarse como `sysContact`, sin el nombre de grupo `system`. Para especificar una búsqueda única de un objeto MIB, porque éste puede aparecer más de una vez, se ingresa el nombre de la MIB seguido por el objeto, tal como `SNMPv2 MIB: sysContact.0`. Para habilitar el acceso aleatorio, se utiliza la opción de línea de comandos `-R`. Esta característica es más útil cuando se busca por objetos MIB que no están localizados en lugares estándar y cuando existe en el agente más de una instancia del mismo nombre de objeto.

Como con cualquier herramienta de software que se comunica con un agente SNMP, debe suministrarse algún método para convertir la notación numérica punteada (como `.1.3.6.1`) del árbol del objeto MIB en la notación que usa nombres (como

`iso.org.dod.internet.mgmt.mib-2`). Normalmente, sin los archivos MIB, las herramientas UCD presentan información MIB que utiliza la forma numérica. Esto se debe a que estas herramientas obtienen sólo la forma numérica del agente; ellas no saben cómo transformar estos identificadores en los nombres de cadena correspondientes. Por ejemplo, cuando se lleva a cabo una `get-request` SNMP en un agente con los archivos MIB no disponibles, puede presentarse el siguiente fragmento de salida:

```
.iso.3.6.1.2.1.1.1.0 = "Cisco Internetwork Operating System
  Software ..IOS (tm) 4500 Software (C4500 J-M), Version
  11.1(5), RELEASE SOFTWARE (fc1)..Copyright (c)
  1986 1996 by cisco Systems, Inc...Compiled Mon 05-Aug-96
  13:17 by mkamson"
.iso.3.6.1.2.1.1.2.0 = OID: .iso.3.6.1.4.1.9.1.50
.iso.3.6.1.2.1.1.3.0 = Timeticks: (99491814) 11 days,
  12:21:58.14
.iso.3.6.1.2.1.1.4.0 = "Neferelle Maxwell"
.iso.3.6.1.2.1.1.5.0 = "remote-gw"
.iso.3.6.1.2.1.1.6.0 = "Remote Sales Office (Florida)"
.iso.3.6.1.2.1.1.7.0 = 78
```

Esta salida proviene de una consulta de un enrutador Cisco, y como se puede observar, la información de ruta del objeto MIB contiene cadenas numéricas sólo después del nombre `iso`. La cadena `iso` se incluyó en la salida porque las herramientas UCD conocen sólo el punto de inicio del árbol MIB. Cuando viene de MIB no estándar o específicos del vendedor, solamente está disponible la forma numérica. Para abordar este asunto, se suministran las opciones `m` y `M`. La opción `m` especifica una lista de módulos MIB que deberían cargarse antes de que la herramienta UCD intente cualquier consulta SNMP en un agente. Cuando se lista más de un módulo MIB, los módulos deberán separarse por dos puntos (:). Un módulo MIB es sólo un archivo que contiene las definiciones MIB para un agente. Usando la opción `m`, se pueden suministrar los módulos correctos MIB, y el fragmento de salida anterior contendrá todos los nombres de cadena. La opción `-M` ayuda, porque se puede suministrar una lista de directorios en la cual se buscan los archivos MIB. Por tanto, el comando

```
# snmpwalk -M /var/mibs cisco-gw1 public system
```

buscará el directorio `/var/mibs`. Suponiendo que encuentra el archivo estándar MIB, presentará lo siguiente:

```
system.sysDescr.0 = Cisco Internetwork Operating System Software
IOS (tm) 4500 Software (C4500-J-M), Version 11.1(5), RELEASE SOFTWARE
(fc1)
Copyright (c) 1986-1996 by cisco Systems, Inc.
Compiled Mon 05-Aug-96 13:17 by mkamson
system.sysObjectID.0 = OID: enterprises.9.1.50
system.sysUpTime.0 = Timeticks: (99487614) 11 days, 12:21:16.14
system.sysContact.0 = Neferelle Maxwell
system.sysName.0 = remote-gw
system.sysLocation.0 = Remote Sales Office (Florida)
system.sysServices.0 = 78
```

Hay también una forma abreviada para especificar todos los módulos MIB (en lugar de suministrar una lista), por medio del comando **all**. Este sobrepone la variable de entorno `MIBS`, la cual se analiza en la siguiente sección.

Variables de entorno

Cada una de las herramientas UCD utiliza un pequeño conjunto de variables de entorno que ayuda a establecer valores globales para ciertos parámetros operativos y atajos para las opciones de línea de comandos. Estas incluyen:

- `PREFIX`
- `MIBS`
- `MIBDIR`
- `SUFFIX`

La variable `PREFIX` suministra una forma estándar para definir el prefijo de los identificadores de objetos MIB. El valor predeterminado es:

```
.iso.org.dod.internet.mgmt.mib-2
```

Si se define esta variable, los contenidos de la variable se agregan al comienzo del objeto MIB que está siendo referenciado cuando se usa una de las herramientas UCD. La definición de este valor ayudará en situaciones en las cuales se utilice una MIB no estándar.

Las variables `MIBS` y `MIBDIR` suministran una forma de cargar módulos MIB adicionales. La variable `MIBS` funciona de la misma manera que la opción de línea de comandos `m`. La variable `MIBDIR` funciona de la misma manera que la opción `M`. Ambas son convenientes, porque trabajan con todas las aplicaciones UCD.

La variable `SUFFIX` activa o desactiva la opción de línea de comandos `s`, la cual presenta el sufijo o el último componente, de la ruta al objeto MIB.

Herramienta snmpdelta

El comando **snmpdelta** recolecta cambios en los valores enteros MIB desde una entidad de agente SNMP. Este comando hace un seguimiento a los objetos enteros especificados y presenta cambios en los objetos que ocurren en el tiempo. Esto es muy útil en el seguimiento de errores del trabajo en red. Puede ser necesario, por ejemplo, para determinar el número de paquetes descartados desde una interfaz, como en el caso del objeto MIB-II `ifInDiscards`. Para hacer un seguimiento a este objeto desde el dispositivo llamado `remote-gw`, puede utilizarse el siguiente comando:

```
# snmpdelta -R remote-gw public ifInDiscards.1
```

Cuando se invoca, este comando producirá lo siguiente:

```
ifInDiscards.1 /1 sec: 0
ifInDiscards.1 /1 sec: 0
ifInDiscards.1 /1 sec: 0
ifInDiscards.1 /1 sec: 0
ifInDiscards.1 /1 sec: 0
ifInDiscards.1 /1 sec: 0
ifInDiscards.1 /1 sec: 0
ifInDiscards.1 /1 sec: 0
ifInDiscards.1 /1 sec: 0
```

Sin la interacción del usuario, el comando anterior sondea continuamente al agente hasta que el usuario utilice un **control+c** (**^c**). Nótese que en el comando que hemos usado, la opción `R` activa el acceso aleatorio al agente MIB. Esto facilita la obtención de los objetos MIB deseados. Una alta tasa de descarte puede indicar problemas con la interfaz. Esta dificultad podría ser causada por un problema de hardware relacionado con el cableado o aun con un error de configuración de software. En el ejemplo anterior, la tasa de descarte no cambió, y esto se debe a que los valores son todos cero. Este comando soporta varios argumentos adicionales de línea de comandos sobre las opciones comunes previamente analizadas. La tabla 8.4 presenta las opciones disponibles.

Tabla 8.4 Opciones de línea de comandos adicionales `snmpdelta`.

Opción	Descripción
`-f`	Lee un archivo de configuración
`-k`	Presenta información de tiempo en la salida
`-l`	Escribe la configuración en un archivo
`-m`	Presenta el valor máximo recuperado
`-p`	Especifica el periodo de sondeo
`-P`	Especifica el periodo de reporte como un número de solicitudes de sondeo
`-s`	Presenta información de sello de tiempo (*timestamp*)
`-S`	Registra datos en un archivo de registro
`-t`	Determina el intervalo de tiempo
`-T`	Produce salida en tabla

Con la opción `m`, el **snmpdelta** presentará el valor máximo obtenido de los objetos MIB sondeados. Presenta los valores de cada sondeo, y cuando se recibe un nuevo valor alto, lo incluye en la columna `Max`. El comando

```
# snmpdelta -R remote-gw public -m ifOutOctets.2
```

produce la siguiente salida:

```
ifOutOctets.2 /1 sec: 25784  (Max: 25784)
ifOutOctets.2 /1 sec: 21287  (Max: 25784)
ifOutOctets.2 /1 sec: 2743   (Max: 25784)
ifOutOctets.2 /1 sec: 8611   (Max: 25784)
ifOutOctets.2 /1 sec: 4473   (Max: 25784)
ifOutOctets.2 /1 sec: 10939  (Max: 25784)
ifOutOctets.2 /1 sec: 1882   (Max: 25784)
ifOutOctets.2 /1 sec: 9258   (Max: 25784)
ifOutOctets.2 /1 sec: 22751  (Max: 25784)
ifOutOctets.2 /1 sec: 28615  (Max: 28615)
ifOutOctets.2 /1 sec: 18599  (Max: 28615)
ifOutOctets.2 /1 sec: 28459  (Max: 28615)
ifOutOctets.2 /1 sec: 2662   (Max: 28615)
```

Nótese que cuando se obtiene el nuevo valor máximo, se actualiza en la columna `Max`, como corresponde.

Cuando se hace un seguimiento a una gran cantidad de datos u objetos MIB, puede ser útil guardar la información en un archivo de registro. La opción `l` guarda los datos en un archivo, en el siguiente formato:

```
{device}-{MIB object}
```

Por tanto, el comando

```
# snmpdelta -R monet public -l ifOutOctets.12
```

registrará deltas `IfOutOctets` en el archivo llamado `monet ifOut-Octets.2`.

La inspección de este archivo revelará el mismo formato y los datos que se presentan normalmente.

Si se necesita presentar la salida de **snmpdelta** en un formato más estructurado, se usa la opción `I`. Por ejemplo, si es necesario hacer un seguimiento a varios objetos MIB y añadir los datos a una hoja de cálculo u otro programa, **snmpdelta** puede presentar la salida en un formato tabular:

```
snmpdelta -R remote-gw public -I ifInDiscards.1 ifInDiscards.2:
ifInDiscards.1 ifInDiscards.2
1592.00 9950.00
9136.00 2506.00
3338.00 5.00
6338.00 2624.00
8665.00 9971.00
5609.00 569.00
9282.00 7086.00
9153.00 8374.00
8653.00 8195.00
9877.00 3827.00
```

La salida es una presentación del número de paquetes descartados de dos interfaces, presentadas en el formato de columna.

Herramienta snmpget

El comando **snmpget** recupera información de una entidad agente SNMP, utiliza una `get request` SNMP con uno o más nombres de objetos MIB como argumentos y retorna sus valores asociados. Si ocurre un error, se mostrará un mensaje descriptivo para ayudar a precisar el problema. Si se especifica una lista de objetos en la línea de comandos, sólo se retornarán aquellos objetos contenidos en la MIB del agente.

La sintaxis del comando es como sigue:

```
snmpget [common arguments] MIB-object [MIB-object]
```

Para recuperar los objetos MIB `sysDescr` y `sysContact` de un enrutador Cisco, puede usarse el siguiente comando:

```
# snmpget remote-gw public system.sysDescr.0 system.sysUpTime.0
```

Dependiendo del modelo y la configuración del enrutador, este comando presentará una salida como la siguiente:

```
system.sysDescr.0 = Cisco Internetwork Operating System Software
IOS (tm) 3000 Software (IGS-INR-L), Version 11.0(17), RELEASE
SOFTWARE (fc1) Copyright (c) 1986-1997 by cisco Systems, Inc.
Compiled Thu 04-Sep-97 14:17 by richv
system.sysUpTime.0 = Timeticks: (134381144) 15 days, 13:16:51.44
```

Nótese que el enrutador Cisco contiene una cadena `sysDescr` bastante larga. Ésta puede ser muy útil cuando se intenta identificar el modelo y la versión actual del sistema operativo de la internetwork (IOS) ejecutado en el sistema. El IOS es el software del sistema Cisco que suministra todas las funciones de enrutado y de procesamiento para un gran conjunto de dispositivos Cisco. En este ejemplo, el modelo del sistema es un enrutador serie 3000 que ejecuta la versión 11.0 que ha estado funcionando durante los últimos 15 días.

Nótese que algunas veces cuando se tiene acceso a objetos de ciertos dispositivos, se pueden encontrar mensajes de error que llevan a creer que el dispositivo en cuestión no soporta objetos especificados con el comando SNMP. Por ejemplo, considérese el siguiente comando que es ejecutado en un sistema Linux llamado `monet`:

```
# snmpget monet private-write system.sysDescr.0
```

El comando produce lo siguiente:

```
system.sysDescr.0 = "Linux monet 2.2.5-15 #5 Mon Aug 2
21:07:04 EDT 1999 i586
```

Sin embargo, ejecutar el mismo comando en el mismo sistema, pero usando una cadena de comunidad diferente, como se muestra:

```
# snmpget monet bad password system.sysDescr.0
```

presenta el siguiente mensaje:

```
Error in packet
Reason: (noSuchName) There is no such variable name in this MIB.
This name doesn't exist: system.sysDescr.0
```

En el ejemplo precedente, el mensaje de error no es claro acerca de qué está sucediendo; el mensaje sugiere que el objeto solicitado no existe en el agente. Obviamente, desde el primer ejemplo, el objeto (`system.sysDescr.0`) existe y tiene un valor asociado. La razón para la disparidad radica en que el agente Linux soporta diferentes vistas del árbol MIB. En el último comando, simplemente intentó buscar el objeto `system.sysDescr.0` en la vista que estaba asociada con la cadena de comunidad `bad password`. Las vistas MIB se introdujeron con el SNMPv2, y el agente Linux soporta estos protocolos. En realidad, como la cadena de comando `bad password` no es válida, el agente emitió la respuesta más apropiada para la situación.

Además, cuando se consultan los dispositivos, como los dispositivos SNMPv1, que no soportan vistas MIB, y el mismo comando se ejecuta en un dispositivo llamado `dips`, se obtiene lo siguiente:

```
Timeout: No Response from dips
```

Este mensaje de error se presenta por la siguiente razón: cuando la cadena de comunidad no corresponde a la contraseña configurada en ésta, un agente SNMPv1 no tiene en cuenta la solicitud. Si se habilitan las trampas de autenticación, entonces envía la trampa asociada. Es importante recordar que cuando se sondea objetos específicos, no debe suponerse que los objetos no están disponibles, a pesar de la generación de mensajes de error que sugieren que no están disponibles. Puede ser útil ejecutar un **snmpwalk** en el agente, si surgen preguntas respecto a los objetos en verdad soportados por el agente.

Herramienta snmpgetnext

El comando **snmpgetnext** recupera uno o más objetos MIB, usando una `getnext request` SNMP. Para cada objeto especificado en la línea de comandos, la `getnext-request` SNMP obtiene el siguiente objeto lexicográfico MIB encontrado en el árbol MIB. Esta herramienta es muy útil para retornar una serie de objetos cuando se desconoce la estructura exacta del objeto MIB que se

está recuperando. Además, permite, de una forma dinámica, el descubrimiento de la estructura de una vista MIB. Por ejemplo, el comando:

```
# snmpgetnext probe public system.sysContact.0
```

obtendrá el siguiente objeto MIB después del objeto `sysContact.0`:

```
system.sysName.0 = "AXON" Hex: 41 58 4F 4E
```

El objeto `sysName.0` se presenta porque está *lexicográficamente próximo* al objeto `sysContract.0`. ¿Cómo estar seguros de esto? Bueno, el enfoque más simple es presentar todo el grupo `system`. Un **snmpwalk** del grupo presenta lo siguiente:

```
system.sysDescr.0 = "AXON LANServant - Ethernet (4.16)"
system.sysObjectID.0 = OID: enterprises.370.2.2
system.sysUpTime.0 = Timeticks: (868306) 2:24:43.06
system.sysContact.0 = "3Com Corporation"
system.sysName.0 = "AXON" Hex: 41 58 4F 4E
system.sysLocation.0 = ""
system.sysServices.0 = 15
```

El propósito principal del comando **snmpwalk** es recuperar una tabla de agente de una manera más efectiva. Considérese la tabla `udp`, la cual contiene información relacionada con cualquier zócalo (*socket*) abierto que use el protocolo de datagrama de usuario. Cuando se sondea esta tabla en un sistema Linux, presenta la salida parcial mostrada en el listado 8.1.

Listado 8.1 Ejemplo de la tabla UDP.

```
1 udp.udpInDatagrams.0 = 860
2  udp.udpNoPorts.0 = 5
3  udp.udpInErrors.0 = 0
4  udp.udpOutDatagrams.0 = 911
5 udp.udpTable.udpEntry.udpLocalAddress.0.0.0.0.111
     IpAddress: 0.0.0.0
6 udp.udpTable.udpEntry.udpLocalAddress.0.0.0.0.161 =
     IpAddress: 0.0.0.0
7 udp.udpTable.udpEntry.udpLocalAddress.0.0.0.0.162 =
     IpAddress: 0.0.0.0
8 udp.udpTable.udpEntry.udpLocalAddress.0.0.0.0.177 =
     IpAddress: 0.0.0.0
```

(*continúa*)

```
 9 udp.udpTable.udpEntry.udpLocalAddress.0.0.0.0.517 =
     IpAddress: 0.0.0.0
10 udp.udpTable.udpEntry.udpLocalAddress.0.0.0.0.518 =
     IpAddress: 0.0.0.0
11 udp.udpTable.udpEntry.udpLocalAddress.0.0.0.0.624 =
     IpAddress: 0.0.0.0
12 udp.udpTable.udpEntry.udpLocalAddress.0.0.0.0.635
     IpAddress: 0.0.0.0
13 udp.udpTable.udpEntry.udpLocalPort.0.0.0.0.111 = 111
14 udp.udpTable.udpEntry.udpLocalPort.0.0.0.0.161 = 161
15 udp.udpTable.udpEntry.udpLocalPort.0.0.0.0.162 = 162
16 udp.udpTable.udpEntry.udpLocalPort.0.0.0.0.177 = 177
17 udp.udpTable.udpEntry.udpLocalPort.0.0.0.0.517 = 517
18 udp.udpTable.udpEntry.udpLocalPort.0.0.0.0.518 = 518
19 udp.udpTable.udpEntry.udpLocalPort.0.0.0.0.624 = 624
20 udp.udpTable.udpEntry.udpLocalPort.0.0.0.0.635 = 635
```

El listado 8.1 incluye objetos *table* que contienen información relacionada, pero no tiene un índice asociado para recuperar cada objeto cuando se necesite. Como se puede observar, los objetos son referenciados por la dirección IP (`0.0.0.0`) empleada, como se muestra en la línea `5`. Esta entrada contiene la IP local y la dirección de puerto para este *socket*. La IP se utiliza como un índice en el objeto listado en la línea `13`, la cual contiene la información del puerto UDP. La habilidad para recuperar objetos MIB basada en un orden lexicográfico es la única manera de descubrir cada objeto en orden.

Herramienta snmpnetstat

El comando **snmpnetstat** es similar a la utilidad `netstat` UNIX y suministra alguna de la misma información básica acerca de las interfaces de dispositivo conectadas y de enrutado. Lo sorprendente de esta herramienta es que suministra una manera fácil de obtener información de interfaz desde cualquier dispositivo SNMP. Esto incluye, por ejemplo, dispositivos como enrutadores, conmutadores, sondas de seguimiento de red y otros dispositivos que soportan el estándar MIB II. Ésta es una herramienta muy poderosa porque la información de interfaz puede recolectarse sin el uso de complicadas secuencias de comandos. Además, evita el uso de interfaces específicas del vendedor cuando una red consta de una gran cantidad de productos de diferentes vendedores.

Como su homólogo de UNIX, el **snmpnetstat** soporta varias opciones de línea de comandos que controlan las operaciones básicas y la salida. La tabla 8.5 presenta las opciones de comando

disponibles. Como se puede observar, muchas de estas opciones son consistentes con aquellas del comando **netstat**.

Tabla 8.5 Opciones de línea de comandos `snmpnetstat`.

Opción	Descripción
`-a`	Muestra el estado de todas las conexiones de *socket*
`-i`	Muestra el estado de todas las interfaces definidas en el sistema
`-l`	Presenta información sobre la interfaz de red especificada
`-o`	Presenta un estado abreviado de interfaces
`-n`	Presenta direcciones de red como números
`-p`	Muestra estadísticas ordenadas por el protocolo de red
`-r`	Presenta información de tabla de enrutado
`-s`	Muestra estadísticas de red por protocolo

Presentación de información de interfaz

Para mostrar la configuración de todas las interfaces de la red, se utiliza la opción `-i`. En el ejemplo siguiente, el comando **snmpnetstat** consulta un enrutador local Cisco llamado `cisco-gw3`:

```
# snmpnetstat -i cisco-gw3 public
Name    Mtu  Network     Address         Ipkts   Ierrs   OpktsOerrs  Queu
Ethern  1500 10.0.2      10.0.2.1        13377   315     132503501   15
Serial  1500 135.111.81  135.111.81.2    431375  127     462082816   50
Serial* 1500 none        none            1934    1544655/336 998     150
```

La salida debería ser familiar; refleja la salida `netstat` UNIX. Las diferencias principales son los nombres de las interfaces y la eliminación de las columnas que representan el número total de colisiones en la interfaz. En general, el software interno Cisco utiliza el mismo tipo de interfaz y el mismo número de índice así como el nombre de interfaz completo (tal como `serial1`). Sin

embargo, en el ejemplo, los nombres están truncados (debido al tamaño de la columna) y se obtienen del tipo de interfaz, no del nombre real de la intrerfaz Cisco, cuando se obtienen del SNMP. Por tanto, cuando se muestra con el comando **snmpnetstat**, el nombre completo de la interfaz no se presenta, ni se muestran los números del índice.

Para listar las interfaces disponibles en una forma abreviada, se utiliza la opción `s`. Como se puede ver a partir de la salida siguiente, sólo se listan las columnas de octetos (bytes) entrantes y salientes. Compárese esto con lo que se presenta con la opción `-i`, en el ejemplo anterior.

```
# snmpnetstat -o cisco-gw4 public
Name    Network      Address         Ioctets   Ooctets
Ethern  10.0.2       10.0.2.1        487708    12778317
Serial  135.111.81   135.111.81.2    4331197   559999
```

El dispositivo `cisco gw4` contiene sólo dos interfaces: una `Ethernet` y una `Serial`. Generalmente, los enrutadores y conmutadores Cisco (y dispositivos de otros vendedores, también) pueden contener un gran número de interfaces. Por ejemplo, la ejecución del comando **snmpnetstat** anterior en un enrutador Cisco 7000 proporcionará lo siguiente:

```
Name     Network    Address    Ioctets     Ooctets
Fddi0/   10.10.1    10.10.1.1  3723440280  1783534532
Fddi1/   10.11.2    10.11.2.1  2560994642  2783361340
Ethern   10.0.2     10.0.2.254 2141819815  1555401237
Ethern*  none       none                   0         0
Ethern*  none       none                   0         0
Ethern*  none       none                   0         0
Ethern*  none       none                   0         0
Ethern*  none       none                   0         0
Fddi3/   10.14.1    10.14.1.1  2248945512 2083011069
Serial   10.250.10  10.250.10.11401691701  870256641
Serial*  none       none       592331671  3226921185
Serial*  none       none                   0         0
Serial*  none       none                   0         0
FastEt   19.80.8    19.82.8.1  4086327200  421590301
FastEt   19.80.9    19.82.9.1  4017448469  3080615899
FastEt   19.80.10   19.82.10.1 269162560   1781784403
FastEt*  none       none                   0         0
```

La cantidad actual de interfaces dependerá del modelo del enrutador y de las tarjetas de interfaz instaladas. En este ejemplo se listaron 17 interfaces. En la salida se listaron cuatro diferentes tipos de interfaz: `Serial`, `Ethernet`, `FastEthernet` y `FDDI`. Esta

salida se sondeó desde un enrutador central vertebral, lo cual explica la alta utilización en muchas de las interfaces.

Para listar una interfaz individual se utiliza la opción `I` seguida del nombre de la interfaz. Esta opción también se utiliza junto con la opción `interval`. Cuando se invoca **snmpnetstat** con el argumento `interval`, muestra un conteo de estadísticas de red relacionadas con la interfaz especificada. La información que se presenta incluye una columna para la interfaz especificada y otra columna que resume la información de todas las otras interfaces. La primera línea de la salida presenta un resumen de información desde que el dispositivo fue reiniciado por última vez. Todas las líneas adicionales representan valores que están cambiando en el intervalo especificado. El comando

```
# snmpnetstat -I Ethernet cisco-gw4 public 10
```

mostrará lo siguiente:

```
input   (Ether)  output  input (Total) output
packets errs packets errs colls packets  errs packets errs colls
68355 39800   131733 198 0499131392464894573 00
178     62    93      68   0    376  255      236     220
46      58    142     84   0    172  167      268     420
93      63    67      60   0    210  134      359     140
119     49    169     85   0    326  187      385     18      0
```

Esta presentación incluye un conteo de actividad del paquete en la interfaz `Etnernet` contenida en el **cisco-gw3**. El comando continuará presentando esta salida hasta que se ingrese un **control+c** (**^c**). Las primeras dos columnas representan la cantidad de paquetes de entrada y de errores de entrada, mientras que las dos siguientes representan la cantidad de paquetes de salida y de errores de salida. La quinta columna suministra el número de colisiones. Las cinco columnas restantes son totales acumulativos para todas las interfaces definidas en el dispositivo.

Presentación de la información de enrutado

Para presentar la información de enrutado desde el mismo dispositivo, se utiliza la opción `-r`, como en el siguiente ejemplo:

```
# snmpnetstat -r cisco-gw3 public
```

La opción `-r` presentará lo siguiente:

```
Routing tables
Destination          Gateway      Flags    Interface
default              161.135.59.1 UG       if0
155.161.75/25        161.135.59.9 U        Serial0
155.161.114.128/26   rembrandt    U        Ethernet0
161.135              161.135.59.1 UG       if0
161.135.59/26        161.135.59.9 U        Serial0
161.135.59.64/26     161.135.59.8 UG       if0
161.135.59.128/26    rembrandt    U        Ethernet0
170.5                161.135.59.1 UG       if0
```

Las estadísticas para cada protocolo de red se soportan con este comando. Por tanto, las opciones `-s` (que muestra estadísticas para cada protocolo) y `-P` (que muestra estadísticas organizadas por cada protocolo) pueden utilizarse para mostrar datos detallados de rendimiento del protocolo. Para ver sólo las estadísticas de cada protocolo, se usa el siguiente comando:

```
# snmpnetstat -s monet public
```

Este comando producirá el siguiente ejemplo de salida:

```
udp:
 8606737 total datagrams received
 7727372 datagrams to invalid port
 1 datagram dropped due to errors
 851929 output datagram requests
tcp:
 0 active opens
 8 passive opens
 0 failed attempts
 1 reset of established connections
 0 current established connections
 645 segments received
 476 segments sent
 0 segments retransmitted
icmp:
 9741 total messages received
 0 messages dropped due to errors
 10042 output message requests
 0 output messages discarded
 Output Histogram:
    Destination unreachable: 310
    Echo Reply: 9732
 Input Histogram:
    Destination unreachable: 9
    Echo Request: 9732
```

```
ip:
 22222667 total datagrams received
 0 datagrams with header errors
 0 datagrams with an invalid destination address
 0 datagrams forwarded
 0 datagrams with unknown protocol
 0 datagrams discarded
 22222673 datagrams delivered
 0 output datagram requests
 0 output datagrams discarded
 0 datagrams with no route
 0 fragments received
 0 datagrams reassembled
 0 reassembly failures
 0 datagrams fragmented
 0 fragmentation failures
 0 fragments created
```

Esta salida ofrece una vista instantánea del rendimiento de la red y de la actividad de cada protocolo de trabajo en red. Muchos de los contadores parecen representar un uso normal de la red. Sin embargo, un valor métrico, `7727372 datagrams to invalid port`, puede representar un problema significativo. Por alguna razón desconocida, los datos están llegando a este sistema, posiblemente de otros sistemas, a uno o más puertos UDP que no son válidos. Una probable causa de esta situación es que una aplicación que debería recibir datos desde otra fuente no está corriendo, pero el otro extremo no lo ha detectado todavía. Otra razón puede ser que una aplicación remota está intentando enviar información a este sistema pero está mal configurada e intentando enviar información a un puerto inexistente.

Una forma de encontrar la causa de este problema es capturar el tráfico en la misma red de este sistema e intentar aprender cuál dispositivo remoto está enviando los datos. Una vez se consigue la identidad del sistema remoto, investigar cuál aplicación está causando este problema es simplemente una cuestión de notificar al dueño del sistema. Es fácil ver por qué el uso del comando **snmpnetstat** es una buena forma de determinar problemas potenciales de red antes de que se salgan de control.

Herramienta snmpset

El comando **snmpset** es uno de los comandos más útiles y poderosos en el paquete UCD. Muchas de las herramientas de este capítulo se enfocan en obtener valores de objetos de un agente

SNMP. Sin embargo, esta herramienta se usa para alterar objetos MIB modificables del agente. La capacidad para alterar un objeto MIB tiene implicaciones profundas, porque al hacerlo así cambia la configuración o el estado operativo de un agente administrado.

Esta herramienta representa un mecanismo poderoso para controlar agentes en una escala global. La capacidad para cambiar la configuración de un gran número de dispositivos suministra una facilidad importante que todo administrador de la red o administrador de sistemas debe tener. Tener el poder para controlar muchos dispositivos puede representar también un riesgo. Considérese, por ejemplo, un enrutador con varias interfaces que sirven como puntos de conexión remota entre importantes redes de oficinas distantes. Un sencillo comando **snmpset** (con la contraseña de seguridad apropiada) ejecutada en una o más interfaces en este enrutador podría inhabilitar la conectividad de la red entre la red local y la(s) oficina(s) remota(s). Obviamente, esto podría tener una consecuencia desastrosa para los negocios, para no hablar de su reputación.

Otro riesgo se relaciona con hacer un cambio global a una serie de sistemas cuando la nueva configuración es incorrecta o causa alguna interrupción del servicio, debido a la naturaleza del cambio. Teniendo en cuenta que el comando **snmpset** tiene mucho poder, se debe ser cauteloso al utilizar este comando en una red activa. Nunca se hará suficiente énfasis en que la primera regla general en el trabajo en red es repasar primero los cambios propuestos en un entorno de prueba. De esa forma, cuando los cambios se realizan en la red real, las fallas y otras sorpresas desagradables se reducen al mínimo.

La sintaxis básica del comando **snmpset** es la siguiente:

```
snmpset [common arguments] MIB-objectID type value [MIB-objectID
type value]
```

`MIB-objectID` es el objeto MIB al que se le dará un nuevo valor. El argumento `type` representa el tipo de objeto que debería alterarse, y `value` representa el nuevo valor del objeto. `Type` es un único carácter sencillo que representa uno de los tipos de objetos listados en la tabla 8.6. El capítulo 2, "Protocolo simple de administración de red", analiza estos tipos de objetos.

Alguien puede preguntar: "¿En qué tareas específicas puede ayudar el comando **snmpset**?" Enseguida se listan algunas de las posibles sugerencias:

- inhabilitar o habilitar una interfaz de red
- actualizar un dispositivo con nueva información de administración (`sysContact`, por ejemplo)
- reiniciar ciertos contadores de tráfico de red
- reiniciar un dispositivo o agente
- modificar algún parámetro de configuración

Tabla 8.6 Tipos de objeto `snmpset`.

Carácter	Tipo de objeto
i	INTEGER
s	STRING
x	HEXADECIMAL STRING
d	DECIMAL STRING
n	NULL OBJECT
o	OBJECTID
t	TIMETICKS
a	IPADDRESS

Se puede recordar un escenario anterior en el cual una interfaz inhabilitada causó un problema en la red. Hay también situaciones cuando el no inhabilitar una interfaz puede causar problemas adicionales en la red. Por ejemplo, durante una tormenta de difusión o cuando un *cracker* intenta penetrar la red, apagar una red o interfaz puede ser la única forma de prevenir que el problema se propague a otras partes de la red. Sin embargo, antes de apagar las interfaces, se debe estar atento por si aparece el síndrome del *conjunto de no retorno*. Considérese el ejemplo de red que se muestra en la figura 8.1.

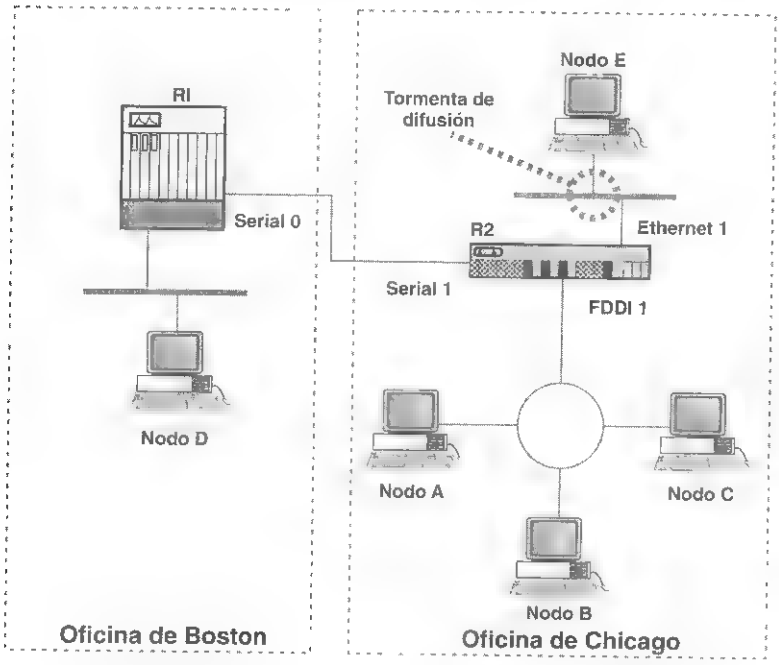


Figura 8.1 Ejemplo del "conjunto de no retorno".

En este ejemplo de red se tienen dos enrutadores, `R1` y `R2`. El único acceso a `R2` es a través del enrutador `R1`, usando la interfaz `Serial1`. Desde el dispositivo `Nodo A`, se emite un comando **snmpset** para inhabilitar la interfaz `serial0` desde `R2`. Esto detiene la tormenta de difusión pero también causa la pérdida de conectividad desde la oficina remota. El problema ahora es: ¿Cómo habilitar la interfaz en `R2` después de arreglar la tormenta de difusión? El conjunto de no retorno significa que los únicos medios para tener acceso al dispositivo han sido cortados. El acceso remoto al enrutador se ha perdido, a no ser que se pueda utilizar alguna capacidad de administración fuera de banda, como la conexión telefónica. En este caso, el puerto debe ser habilitado desde el lado local, lo cual podría requerir instruir a alguien para habilitar el puerto o reiniciar el dispositivo para reestablecer el puerto.

La mejor forma de evitar el *conjunto de no retorno* es inhabilitar la interfaz local `serial1` en `R1` o la interfaz remota `ethernet1` en `R2`. De esta forma, se controla el problema de difución y aún se mantiene el acceso a los dispositivos.

Para inhabilitar una interfaz `serial0` en el enrutador `R1`, debe hacerse una modificación del objeto `IfOperStatus`. Esto incluye establecer el objeto en cero (0) para inhabilitar la interfaz desde una perspectiva administrativa. Este objeto se establece usando el siguiente comando:

```
# snmpset monet private interfaces.ifTable.ifEntry.ifAdminStatus.2 i 0\
interfaces.ifTable.ifEntry.ifAdminStatus.3 = 0
```

Cuando se ejecuta, el comando inhabilita el flujo de tráfico hacia esta interfaz y desde ella. En la mayoría de los agentes, este cambio del objeto no es permanente, lo que quiere decir que un reinicio del sistema establecerá el valor del objeto al valor predefinido (habilitado). Sin embargo, sin una reiniciación del sistema, la única otra forma de restaurar la conectividad de la red será habilitando la interfaz mediante el comando **snmpset**.

El comando **snmpset** puede alterar otros objetos MIB en el agente. Por ejemplo, considérense los objetos `sysContact` y `sysLocation`. Usando un simple comando **snmpset** se asignan nuevos valores a estos objetos.

```
# snmpset cisco-gw10 private system.sysContact.0 s "NCC Support"\
system.sysLocation.0 s "3rd Floor Closet"
```

Cuando el comando se ha ejecutado, retorna lo siguiente:

```
system.sysContact.0 = "NCC Support"
system.sysLocation.0 = "3rd Floor Closet"
```

Herramienta snmpstatus

El comando **snmpstatus** obtiene información importante desde una entidad de red SNMP mediante la operación `get-request` SNMP. La sintaxis del comando es la siguiente:

```
snmpstatus host community
```

Cuando el agente remoto utiliza la cadena genérica de comunidad `public`, ésta puede omitirse de los comandos SNMP, porque `public` se utiliza de manera predeterminada. Por cierto, no se encomienda el uso de la cadena de comunidad `public`.

Cuando el comando **snmpstatus** se utiliza en un dispositivo, éste presenta la siguiente información:

- dirección IP del dispositivo
- objeto MIB `sysDescr`

- objeto MIB `SysUpTime`
- cantidad de paquetes recibidos y transmitidos en todas las interfaces activas (es decir, la suma de los objetos `ifInUCastPkts.*` e `ifInNUCastPkts.*`)
- cantidad de paquetes IP recibidos y transmitidos (es decir, `ipInReceives.0` e `IpOutRequests.0`)
- cantidad de interfaces activas
- número de interfaces inhabilitadas

Por tanto, al ejecutar el comando

```
# snmpdtatus switcn 2200 private
```

en un dispositivo llamado switch-2200, se presenta la salida que se muestra en la figura 8.2.

Usar este comando es una buena forma de obtener una vista instantánea de un dispositivo de red, sin conocer mucho acerca del nodo o de algunos de los objetos MIB específicos que se van a sondear. Como se puede observar, el dispositivo es `Linkswitch 2200` (fabricado por 3Com), corriendo la versión `7.1` del software del conmutador. El dispositivo ha estado corriendo durante `90` horas y `23` minutos. Nótese que **snmpstatus** ha detectado ocho interfaces que están abajo o desactivadas. Esto no es un problema en este caso porque el dispositivo es un conmutador Ethernet donde las interfaces están conectadas a computadoras personales o a estaciones de trabajo.

Algunos usuarios reinician sus sistemas más a menudo y apagan sus sistemas cuando dejan la oficina cada noche. Esto causa que los puertos del conmutador sean marcados como apagados, y explica por qué se muestran arriba en la salida **snmpstatus**.

Esta herramienta puede hacer un inventario rápido de una red por medio del uso de shell. Por ejemplo, considérese el siguiente comando:

```
# foreach node (10.0.2.1 10.0.2.202 10.0.2.254 10.0.2.50)\
foreach? snmpstatus $node public; echo
"************************"
foreach? end
```

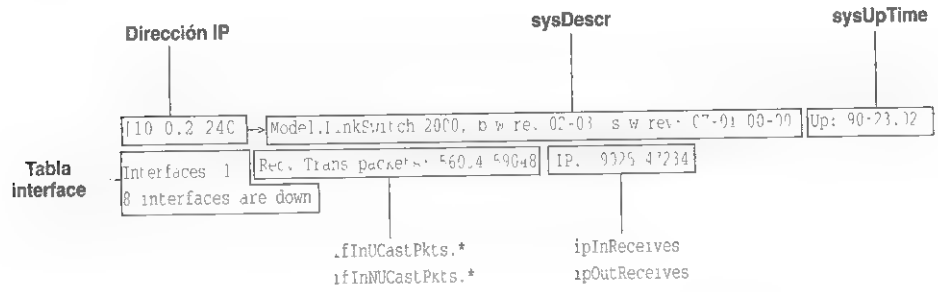


Figura 8.2 Salida del comando `snmpstatus`.

Este comando genera la siguiente salida:

```
[10.0.2.1]=>[Cisco Internetwork Operating System Software
IOS (tm) GS Software (RSP-JV M), Version 11.1(16)CA, EARLY
  DEPLOYMENT RELEASE SO
FTWARE (fc1)
Synced to mainline version: 11.1(16)
Copyright (c) 1986 1997 by cisco Systems, Inc.
Compiled Sat 20-Dec-97 04:21 by] Up: 168 days, 3:23:39
Interfaces: 23, Recv/Trans packets: 1270803594/ 666101155 | IP:
  1516575442/102
93/871
9 interfaces are down!
**********************
[10.0.2.202]=>[Cisco Systems WS-C5000] Up: 167 days, 15:31:32
Interfaces: 100, Recv/Trans packets: 258062534/866645 | IP:
  22222133/0
32 interfaces are down!
**********************
[10.0.2.254]=>[Fibre Channel Switch.] Up: 1 day, 8:35:11
Interfaces: 2, Recv/Trans packets: 43996343/16638 | IP:
  20002011/7886
**********************
[10.0.2.50]=>[AXON LANServant - Ethernet (4.16)] Up: 9:45:21
Interfaces: 3, Recv/Trans packets: 4027/1135 | IP: 0/0
```

Usando el comando incorporado **foreach** del `csh`, se asigna una lista de direcciones IP entre paréntesis. Cuando se ejecuta **foreach**, éste hace un llamado al comando **snmpstatus** y remplaza cada una de la direcciones IP con la variable `node`. En la salida anterior, es obvio que esta herramienta es muy fácil de usar para sondear la red. En la práctica, sería más razonable construir una secuencia de comandos shell para hacer un inventario de estos dispositivos sobre una base más continua, en lugar de hacerlo

interactivamente desde la línea de comandos. Además, la salida podría colocarse en un archivo de texto para una revisión posterior.

Herramienta snmptable

El comando **snmptable** sondea un agente y presenta una tabla MIB, usando la operación `get next request` SNMP. La sintaxis del comando es:

```
snmptable [common options ] [additional options] host community tableID
```

Los argumentos mínimos requeridos incluyen `host`, `community` y `tableID`. El argumento `tableID` debe ser una tabla real SNMP, como `interfaces.ifTable`. La tabla `ifTable` contiene una serie de objetos MIB que proporcionan información de rendimiento y otras características de las interfaces en un dispositivo. En la MIB-II, la cual se describe en el capítulo 6, "Vista general de MIB-II", se definen y pueden mostrarse con este comando las siguientes tablas:

- tabla de interfaz (`interfaces.ifTable`)
- tabla de direcciones IP (`ip.ipAddrTable`)
- tabla de enrutado IP (`ip.ipRouteTable`)
- tabla de medios de red IP (`ip.ipNetToMediaTable`)
- tabla de conexión TCP (`tcp.tcpConnTable`)
- tabla del oyente UDP (`udp.udpTable`)

Nótese que el capítulo 6 describe el árbol MIB-II. El propósito básico de este comando es dar al usuario la capacidad de presentar tablas SNMP e importar los datos en otros programas para reporte y manipulación adicionales. Por ejemplo, para presentar la tabla de conexión TCP, `tcpConnTable`, se utiliza el siguiente comando:

```
# snmptable 10.0.2.240 public tcp.tcpConnTable
SNMP table: tcp.tcpConnTable
tcpConnState tcpConnLocalAddress  tcpConnLocalPort tcpConnRemAddress
tcpConnRemPort
listen       0.0.0.0      23           0.0.0.0           0
listen       0.0.0.0      111          0.0.0.0           0
listen       0.0.0.0      513          0.0.0.0           0
established  10.0.2.240   23           10.0.2.75         33441
```

Si se suministra un objeto diferente a una tabla, se generará un error. Por tanto, el comando

```
# snmptable 10.0.2.240 public system.sysContact.0
```

retornará

```
Was that a table? system.sysContact.0.1.1
```

En este caso se proporciona el objeto MIB, `system.sysContact.0` (el cual no es una tabla MIB, definitivamente), con el comando `snmptable`. Como se puede observar, se presenta un error apropiado, que incluye cuál debe ser el primer elemento en la tabla (es decir 1.1).

El comando **snmptable** soporta unos pocos argumentos adicionales de línea de comandos que pueden controlar el formato de la salida. La tabla 8.7 presenta estas opciones adicionales.

Tabla 8.7 Opciones adicionales `snmptable` de la línea de comandos.

Opción	Descripción
`-b`	Presenta un encabezamiento breve
`-f`	Especifica un carácter separador
`-h`	Evita la presentación de información de encabezamiento
`-H`	Presenta solamente información del encabezamiento de la tabla
`-w`	Especifica el ancho de la tabla que se está presentando
`-x`	Añade el índice MIB al comienzo de cada entrada listada

Una de las opciones más útiles es `f`, la cual se utiliza para especificar un carácter separador alternativo de columna. Por tanto, al emplear los dos puntos (:) como el separador de campo cuando se presenta la misma tabla TCP anterior, se usa la siguiente secuencia de comandos (mostrada con la salida):

```
# snmptable 10.0.2.240 public -f : tcp.tcpConnTable
SNMP table: tcp.tcpConnTable
tcpConnState:tcpConnLocalAddress:tcpConnLocalPort:tcpConnRe
mAddress:tcpConnRemPort
listen:0.0.0.0:23:0.0.0.0:0
listen:0.0.0.0:111:0.0.0.0:0
listen:0.0.0.0:513:0.0.0.0:0
established:10.0.2.240:23:10.0.2.75:33441
```

Cada una de las columnas está ahora separada por el carácter :, facilitando el análisis de esta información si se utiliza como entrada en otros programas. La opción `-b` puede utilizarse para suministrar una etiqueta de columna más descriptiva, como se muestra aquí. La tabla `at.atTable` contiene la traducción de direcciones transformadas entre direcciones IP y direcciones físicas. En este caso, se presentan tres etiquetas de columna: `IfIndex`, `PhysAddress` y `NetAddress`. `IfIndex` es el índice en la tabla de interfaz y muestra desde cuáles interfaces se originaron estas transformaciones. `PhysAddress` es la dirección del enlace de datos (Ethernet en este caso) y `NetAddress` es la dirección IP asociada con cada dispositivo en la red. El comando:

```
# snmptable cisco-gw5 public  b at.atTable
```

sondea al grupo de traducción de dirección MIB-II y suministra la siguiente salida de ejemplo:

```
SNMP table: at.atTable
 IfIndex           PhysAddress   NetAddress
       2 "08 00 20 8D 81 82 "    10.0.1.100
       2 "00 60 47 1F 76 8A "    10.0.1.120
      14 "00 60 08 91 4A 64 "    10.0.2.110
      14 "00 60 95 AD F2 1F "    10.0.2.111
      14 "00 80 5F E2 8B 3D "    10.0.2.117
      14 "00 60 08 94 37 FC "    10.0.2.200
      14 "08 00 20 86 2F C2 "    10.0.2.221
      14 "00 60 08 62 C7 3E "    10.0.2.226
      14 "08 00 20 7A CA 49 "    10.0.2.234
      14 "08 00 20 25 70 E7 "    10.0.2.250
      14 "00 20 AF CA E6 99 "    10.0.2.251
      14 "00 60 08 1D FD 4D "    10.0.2.252
      14 "00 80 A3 03 4E 9B "    10.0.2.257
```

Herramienta snmptest

El comando **snmptest** suministra una facilidad simple parecida a shell, que le facilita comunicarse con una entidad de red que usa SNMP. A diferencia de muchas otras herramientas SNMP, la **snmptest** es interactiva, solicitando información para cumplir una o más tareas.

El software soporta tres modos de operación: `snmp-get`, `snmp getnext` y `snmp-set`. Emitir el comando **$N** colocará a **snmptest** en el modo `snmp getnext`, mientras que usar **$S** pone al comando en el modo `snmp set`. Para volver al modo `snmp-get`, se utiliza el comando **$G**. **snmptest** se coloca en el modo `snmp get` de manera predeterminada, y en este modo al usuario se le solicita un objeto MIB a recuperar de un dispositivo SNMP.

Por ejemplo, la siguiente secuencia de comandos obtendrá al objeto `sysUpTime` de un sistema llamado `monet`:

```
# snmptest monet stevewrite
Variable: system.sysUpTime.0
Variable:
Received Get Response from monet
requestid 0x4362F60A errstat 0x0 errindex 0x0
system.sysUpTime.0 = Timeticks: (379613) 1:03:16.13
Variable: ^c
```

Se puede notar claramente que ésta es una interfaz rudimentaria para emitir operaciones básicas SNMP hacia los agentes de red. Esto es útil si se debe ejecutar una gran cantidad de tareas en el mismo objetivo de red. Nótese que para salir del programa, simplemente se utiliza la secuencia de comando **^c**.

Herramienta snmptranslate

La herramienta **snmptranslate** traduce los objetos MIB SNMP a un formato más amigable y legible. Cuando este comando se ejecuta con un objeto MIB, éste traducirá al objeto en un valor SMI o en una forma simbólica. Cuando no se especifica ninguna opción, presenta de manera predeterminada el valor SMI. La tabla 8.8 incluye las opciones de línea de comandos soportadas con esta herramienta. Este comando se usa principalmente para ayudar a presentar las características completas de los objetos MIB, sin recurrir a la lectura de los archivos apropiados de definición MIB.

Tabla 8.8 Opciones de línea de comandos **snmptranslate**.

Opción	Descripción
`-d`	Presenta una descripción del objeto
`-p`	Presenta la tabla de símbolos desde los archivos MIB cargados
`-n`	Presenta objetos en formato simbólico
`-s`	Presenta sólo la última parte simbólica de la OID
`-R`	Utiliza acceso aleatorio cuando accede los objetos
`-w`	Presenta advertencias cuando ocurren conflictos de símbolos
`-W`	Presenta advertencias más expresivas que las que presenta `-w`

snmptranslate presenta la notación numérica punteada del objeto de manera predeterminada. Por tanto, la traducción del objeto MIB `system.sysDescr` puede llevarse a cabo con el comando:

```
# snmptranslate system.sysDescr
```

el cual genera lo siguiente:

```
.1.3.6.1.2.1.1.1
```

Para presentar una descripción bastante completa del objeto MIB, se utiliza la opción `-d`. Por ejemplo, considérese el siguiente comando:

```
# snmptranslate -d system.sysDescr
```

Cuando se ejecuta, el comando produce lo siguiente:

```
.1.3.6.1.2.1.1.1
SYNTAX OCTET STRING
DISPLAY-HINT "255a"
MAX-ACCESS   read-only
```

(*continúa*)

```
STATUS current
DESCRIPTION "A textual description of the entity.
This value should include the full name and version
identification of the system's hardware type, software
operating-system, and networking software."
```

Como se mostró, puede obtenerse una referencia completa del objeto MIB. Esta información se toma de los archivos de definición MIB que el agente referencia. En este caso, se utilizó el archivo `RFC1213-MIB.txt`, el cual se localiza en el directorio `/esr/local/share/mibs`.

Herramienta snmptrap

El comando **snmptrap** emitirá una trampa SNMP para uno o más administradores SNMP designados. Esta herramienta es muy útil cuando está insertada en una secuencia de comandos shell u otro programa para enviar trampas. Uno o más identificadores de objetos pueden especificarse en la línea de comandos, además el tipo y el valor deben acompañar a cada objeto. **snmptrap** soporta tanto al formato SNMPv1 como al formato SNMPv2. La sintaxis básica del comando es:

```
snmptrap -v 1 [command arguments] enterprise oid agent generic-trap\
specific-trap uptime [object ID type value]
```

El campo `enterprise-oid` identifica al subsistema de administración de la red que generó la trampa. El `agent` es el *host* que emite la trampa. `generic-trap` corresponde a una de las trampas SNMP predefinidas listadas en el capítulo 2, "Protocolo simple de administración de red". El valor `specific trap` indica más específicamente la naturaleza de la trampa. El campo `uptime` se utiliza como un sello de tiempo entre la última inicialización del dispositivo y la emisión de la trampa. `objectID`, `type` y `value` suministran información adicional relacionada con la trampa. Estos campos adicionales se conocen como el vínculo variable, y pueden contener cualquier tipo de información relacionada con la trampa.

Los campos `enterprise-oid`, `agent` y `uptime` no necesitan especificarse en la línea de comandos. En cambio, la secuencia de caracteres vacía `''` puede usarse para especificar los valores predeterminados para estos campos. El valor predeterminado del agente es el nombre del *host* de la máquina donde está corriendo el comando **snmptrap**. `uptime` se obtiene del objeto MIB `system.sysUpTime.0` del sistema local.

Considérese, por ejemplo, que se desea gustaría emitir una trampa `link-down` a un sistema de administración de la red llamado `rembrandt`. Además, si se quiere comunicar que un puerto particular se apagó, se incluye el puerto en el vínculo variable de la trampa. Se puede usar el siguiente comando:

```
# snmptrap -v 1 public '' monet 2 0 '' interfaces.iftable.ifentry.\
ifindex.1 i 1
```

En este ejemplo, se utiliza el valor predeterminado para los campos `enterprise-oid` y `uptime`. Además, se especifica la interfaz particular (`ifIndex-1`) y se establese el valor en 1 (el cual indica la segunda interfaz con el dispositivo). El 2 representa la trampa `link down` y el 0 suministra un valor nulo para el valor `specific-trap`.

Si repasamos las trampas en `rembrandt`, veremos una salida como la siguiente:

```
Nov 24 17:51:27 monet snmptrapd[385]: 10.0.2.201: Link Down
Trap (0) Uptime:2:26:59, interfaces.ifTable.ifEntry.
  ifIndex.1 = 1
```

En la práctica, la mayoría de mensajes `link down` no son así de significativos o interesantes. Sin embargo, cuando vienen de dispositivos críticos, como enrutadores centrales o conmutadores, una interfaz inhabilitada podría significar un desastre para la red.

Debe tenerse cuidado cuando se configuran trampas de los dispositivos de la red, porque puede causar una *inundación de trampas* (*trap flood*). Esta condición ocurre cuando se envía un gran número de trampas, en respuesta a un evento recurrente en particular. Por ejemplo, en un conmutador LAN Ethernet, un estado de enlace de puerto puede rebotar de arriba abajo muchas veces por segundo, debido a un mal funcionamiento del sistema. Este tipo de problema puede generar una gran cantidad de trampas. En la práctica, una notificación de trampa debería habilitarse solamente en dispositivos y servicios críticos. En este caso, si es posible, las trampas deberían inhabilitarse para eventos de conexión y desconexión de dispositivos genéricos de la red.

Herramienta snmptrapd

El comando **snmptrapd** recibirá y registrará trampas SNMP. Las trampas que se envían al puerto 162 se registran en la facilidad

syslog UNIX o se presentan en el terminal. Estos mensajes se envían usando los niveles de registro LOG_WARNING y LOG_LOCAL0. El comando **snmptrapd** debe ejecutarse como un superusuario, debido a que escucha en un puerto reservado del sistema. Ejecutar el comando sin alguna de las opciones lo colocará en un segundo plano y lo desvinculará del shell.

El comando **snmptrapd** soporta varios argumentos de línea de comandos: P, D, -d y q. La opción P instruirá al **snmptrapd** para que presente cualquiera de las trampas recibidas en la salida estándar, y la opción -d presentará un vaciado del paquete de trampa. Por tanto, para mostrar trampas recibidas y presentar los contenidos de los paquetes de trampa, se utiliza el siguiente comando:

```
# snmptrapd -P -d
```

Si se ejecuta este comando usando el ejemplo previo sobre el *host* rembrandt, se presenta la siguiente salida:

```
1999-11-27 22:56:47 UCD-snmp version 4.0.1
received 69 bytes  from 10.0.3.126:-32566:
0000: 30 82 00 41  02 01 00 04  06 70 75 62  6C 69 63 A4
  0..A.....public.
0016: 82 00 32 06  08 2B 06 01  04 01 03 01  01 40 04 0A
  ..2..+.......@..
0032: 00 03 7E 02  01 02 02 01  00 43 03 24  5C 96 30 82
  ..~.......C.$\.0.
0048: 00 13 30 82  00 0F 06 0A  2B 06 01 02  01 02 02 01
  ..0.....+.......
0064: 01 01 02 01  01                                    .....
1999-11-27 22:56:51 monet [10.0.3.126] enterprises.3.1.1:
      Link Down Trap (0) Uptime: 6:37:09
      interfaces.ifTable.ifEntry.ifIndex.1 = 1
```

La primera línea muestra la versión actual, y cuándo se ejecutó el proceso **snmptrapd**. A continuación, la salida presenta el número de bytes en el paquete de trampa, el *host* desde donde se envió la trampa (monet) y el puerto UDP (32566, en este caso). Los contenidos de trap incluyen tanto caracteres hexadecimales como caracteres ASCII. Finalmente, se muestra la información de trap, incluida la información de vínculo variable.

La opción -D presentará información adicional de depuración, que incluye el análisis de archivos MIB. La opción q suministra una salida más expresiva, pero no parece estar implementada en esta versión del comando.

El comando **snmptrapd** puede usarse como un punto focal para la recepción de trampas de una gran cantidad de dispositivos. Sin embargo, debe estar corriendo continuamente para recibir trampas desde la red. De manera predeterminada, esta herramienta no es arrancada por el sistema durante el inicio del mismo. Para habilitar este comando en el arranque del sistema, debería agregarse a la secuencia de comandos de inicio del agente UCD llamada `snmpd`. Esta secuencia de comandos shell puede encontrarse en el directorio `/etc/rc.d/init.d`. El listado 8.2 muestra la secuencia de comandos de inicio con el comando **snmptrapd** añadido en la línea número 21.

Listado 8.2 Secuencia de comandos de inicio UCD con el comando **snmptrapd.**

```
 1 #!/bin/bash
 2
 3 # ucd-snmp init file for snmpd
 4 #
 5 # chkconfig: - 50 50
 6 # description: Simple Network Management Protocol (SNMP) Daemon
 7 #
 8 # processname: snmpd
 9 # config: /etc/snmp/snmpd.local.conf
10 # config: /etc/snmp/snmpd.conf
11 # config: /usr/share/snmp/snmpd.local.conf
12 # config: /usr/share/snmp/snmpd.conf
13
14 # source function library
15 . /etc/rc.d/init.d/functions
16
17 case "$1" in
18    start)
19          echo -n "Starting snmpd: "
20          daemon /usr/sbin/snmpd -l /etc/snmp/agent.log
21          daemon /usr/sbin/snmptrapd -s
22          touch /var/lock/subsys/snmpd
23          echo
24          ;;
25    stop)
26          echo -n "Shutting down snmpd: "
27          killproc snmpd
28          rm -f /var/lock/subsys/snmpd
29          echo
30          ;;
31    restart)
```

(*continúa*)

```
32          $0 stop
33          $0 start
34          ;;
35    status)
36          status snmpd
37          ;;
38    *)
39          echo "Usage: snmpd {start|stop|restart|status}"
40          exit 1
41 esac
42
43 exit 0
```

En este ejemplo, la secuencia de comandos se ejecutará en el arranque del sistema y el comando **snmptrapd** se invocará con la opción `-s` (*véase* la línea 21). Esta opción habilita el registro de mensajes a la facilidad `syslog`. Antes de que `syslog` pueda recibir los mensajes, sin embargo, debe actualizarse el archivo de configuración `/etc/syslog.conf` para incluir una entrada para el procesamiento de trampa. El listado 8.3 muestra el `syslog.conf` Linux estándar con la entrada de trampa apropiada agregada (*véase* la línea 25).

Listado 8.3 Archivo `syslog.conf` estándar con entrada **trap**.

```
1 # Log all kernel messages to the console.
2 # Logging much else clutters up the screen.
3 #kern.*
  /dev/console
4
5 # Log anything (except mail) of level info or higher.
6 # Don't log private authentication messages!
7 *.info;mail.none;news.none;authpriv.none
  /var/log/messages
8
9 # The authpriv file has restricted access.
10 authpriv.*
   /var/log/secure
11
12 # Log all the mail messages in one place.
13 mail.*
   /var/log/maillog
14
15 # Everybody gets emergency messages, plus log them on
   another
16 # machine.
17 *.emerg*
```

(*continúa*)

```
18
19 # Save mail and news errors of level err and higher in a
20 # special file.
21 uucp,news.crit
   /var/log/spooler
22
23 # Save boot messages also to boot.log
24 local7.*
   /var/log/boot.log
25 local0.*
   /var/log/trapd.log
26
27 #
28 # INN
29 #
30 news.=crit
   /var/log/news/news.crit
31 news.=err
   /var/log/news/news.err
32 news.notice
   /var/log/news/news.notice
```

La nueva línea define la entrada `syslog` que el **snmptrapd** utiliza cuando se envían los mensajes al proceso `syslogd`. La entrada indica que se recolectarán todos los mensajes de la categoría `local0`. Nótese que todos los mensajes obtenidos se guardarán en el archivo `/var/log/trapd.log`.

Herramienta snmpwalk

El comando **snmpwalk** *recorrerá* un árbol MIB del agente usando la `getnext-request` SNMP ¿Por qué usar este comando, en lugar de la herramienta **snmpget**? Porque el comando **snmpwalk** puede descubrir automáticamente todo el almacén MIB contenido en el agente. Con **snmpget**, se necesita referenciar explícitamente un objeto para obtener un valor. Con el comando **snmpwalk**, se puede arrancar en un punto dado y moverse a través de la MIB del agente hasta el final. En la línea de comandos puede darse una variable objeto para especificar con cuál porción del espacio MIB empezará la búsqueda. Sin un argumento de objeto MIB, el **snmpwalk** busca y arranca con el almacén de objeto MIB-II. Por tanto, el comando

```
# snmpwalk monet public
```

recorrerá todo el agente del sistema en el dispositivo monet. Puesto que la MIB-II contiene una gran cantidad de objetos, el comando producirá un listado bastante grande. En lugar de listar cada objeto soportado por el agente, se puede limitar la búsqueda y presentar sólo un grupo sencillo de objetos. Por tanto, se pueden listar todos los objetos hallados en el grupo system MIB-II usando el siguiente comando:

```
# snmpwalk cisco-gw5 public system
```

Ejecutar este comando en el enrutador Cisco genera la siguiente salida de ejemplo:

```
system.sysDescr.0 = Cisco Internetwork Operating System Software
IOS (tm) GS Software (RSP-JV-M), Version 11.1(13a)CA1, EARLY
DEPLOYMENT RELEASE SOFTWARE (fc1)
Synced to mainline version: 11.1(13a)
Copyright (c) 1986-1997 by cisco Systems, Inc.
Compiled Wed 13-Aug-97 04:12 by richardd
system.sysObjectID.0 = OID: enterprises.9.1.46
system.sysUpTime.0 = Timeticks: (236153209) 27 days,
7:58:52.09
system.sysContact.0 = Susan Maxwell
system.sysName.0 = cisco-gw4
system.sysLocation.0 = Testing Lab
system.sysServices.0 = 78
```

Para recorrer toda la MIB en un agente dado y guardar la salida en un archivo, se utiliza el siguiente comando:

```
# snmpwalk cisco-gw5 public .1 > walk.out
```

Este comando utiliza .1 como un punto de inicio para empezar a listar los objetos. Hacer esto asegurará que se presenten todos los objetos porque .1 es la raíz de todo el árbol MIB y todos los objetos son accesibles desde este punto. Recorrer todo el árbol MIB con un agente ayudará a identificar ciertos objetos MIB o dará una idea exacta de cuántos objetos puede soportar un agente en particular. Para ver el número aproximado de objetos que soporta el cisco-gw5, se cuenta el número de líneas en el archivo. Debido a que el comando **snmpwalk** presenta cada objeto MIB en su propia línea (a no ser que la línea sea más larga que el número máximo de caracteres por línea), se puede usar entonces el comando UNIX **wc** para totalizar el número de líneas en el archivo walk.out. Por tanto, el comando

```
# wc -l walk.out
```

produce la siguiente salida:

```
2242 walk.out
```

Esta salida de comando muestra que el agente contenido en el dispositivo `cisco-gw5` soporta aproximadamente 2242 objetos MIB. Cuando no se especifica ningún objeto con el comando **snmpwalk**, éste buscará de manera predeterminada el árbol de objetos MIB-II. Cuando el comando **snmpwalk** alcanza el final de la MIB en el agente, presentará el mensaje `End of MIB`.

Nótese que el uso de **snmpwalk** es un medio bastante ineficiente para obtener un gran número de objetos MIB de un agente. Esto se debe a que el comando consulta constantemente un solo objeto MIB para obtener un valor asociado. Se recomienda usar el comando **snmpbulkwalk** cuando sea posible, para reducir el tráfico de la red y la carga en el sistema del agente. Este comando reduce de manera significativa el trabajo en un nivel inferior para obtener una gran cantidad de información de un agente. *Véase* la sección **snmpbulkwalk** para obtener información adicional.

Herramienta snmpbulkwalk

La herramienta **snmpbulkwalk** se comunica con una entidad de red usando una `bulk request` SNMPv2. Como la herramienta **snmpwalk**, la **snmpbulkwalk** recorrerá un árbol MIB hasta alcanzar el final de la MIB o hasta que ocurra un error. La `bulk request` suministra un mecanismo más eficiente para transferir una gran cantidad de datos que la *get-request* SNMP v1 regular. Por ejemplo, suponga que se quiere recuperar todo el grupo `interface` MIB-II de un servidor Linux. Se emitiría el siguiente comando:

```
# snmpwalk monet public interfaces
```

Este comando generará una serie de `getnext-requests` SNMP para el grupo `interfaces` MIB en un nodo de red llamado `monet`. El comando **snmpwalk** SNMPv1 (el cual usa un `get-request` normal) obtendrá la información usando 136 paquetes. ¿Cómo se sabe esto? Justo antes de ejecutar el comando **snmpwalk**, se inició una captura de paquete, usando el siguiente comando:

```
# tcpdump host monet -w output.file
```

El comando **tcpdump**, el cual se describe en el capítulo 5, "Utilidades y herramientas adicionales del sistema", capturará todos los paquetes que vienen desde el *host* llamado `probe` o se dirigen a éste. Estos paquetes se guardan en el archivo llamado `output.file`. Una vez el comando **snmpwalk** ha terminado, el comando **tcpdump** se detiene de manera manual. A continuación, simplemente se cuenta el número de paquetes capturados en el archivo `output.file`. Por tanto, con

```
# tcpdump  r output.file | wc -l
     136
```

se observa que se capturó un total de 136 paquetes. Éste es el número de paquetes necesarios para recuperar la tabla `interfaces`. Nótese que, dependiendo del número de interfaces definidas en un dispositivo, se requerirán más o menos paquetes. Sin embargo, si se consulta el mismo dispositivo usando el comando **snmpbulkwalk**,

```
# snmpbulkwalk -v 2c monet public interfaces
```

se sondeará la misma información del grupo de interfaces ¡pero sólo se usarán dos paquetes!

Éste es un gran ahorro en términos de ancho de banda de red y un incremento en el rendimiento SNMP, tanto para el agente como para el administrador. La opción *bulk-request* SNMP es eficiente porque intenta colocar tantos valores OID MIB como sea posible en el campo del vínculo variable de el/los paquete(s) de respuesta SNMP. El único límite para la cantidad de datos que pueden colocarse en el paquete SNMP es el tamaño máximo de paquete de los protocolos. Por cierto, el **tcpdump** no es el único comando disponible para capturar el tráfico de la red. También puede usarse la herramienta **ethereal**, descrita en el capítulo 5, "Utilidades y herramientas adicionales del sistema". La figura 8.3 contiene la salida de la captura de la operación **snmpbulkwalk** usando el software Ethereal. Como se puede observar en la figura, sólo se capturarán dos paquetes cuando se ejecutó el comando **snmpbulkwalk**: la `bulk-request` SNMP y la respuesta. En el ejemplo, el paquete de respuesta contiene todos los objetos MIB de la tabla `interfaces`. Esto explica por qué este paquete es significativamente más grande que los paquetes de respuesta de la operación **snmpget** o de la operación **snmpgetnext**.

Para recorrer todo el árbol MIB del dispositivo llamado `cisco-gw1`, se utiliza el siguiente comando:

```
# snmpbulkwalk -v 2c cisco-gw1 public
```

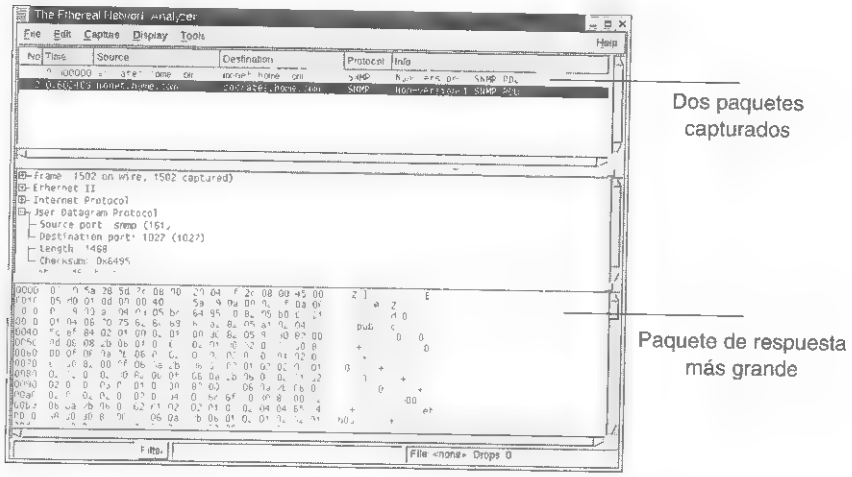


Figura 8.3 Captura del paquete de operación `snmpbulkwalk`.

El comando **snmpbulkwalk** sólo trabaja con los agentes SNMPv2 o SNMPv3. Esto se debe a que la especificación de la facilidad `bulk request` vino después del uso generalizado del SNMPv1. Por tanto, el uso de este comando en un agente SNMPv1 generará un error.

Por ejemplo, el siguiente comando intenta recorrer el árbol MIB de un dispositivo llamado `remote gw` empezando desde el grupo `system`. En este ejemplo, `remote gw3` soporta sólo el SNMPv1:

```
# snmpbulkwalk -v 2c remote-gw3 public system
```

Como este dispositivo no entiende el `bulk request` SNMP, eventualmente expira, y se presenta el siguiente mensaje después del comando **snmpbulkwalk**:

```
Timeout: No Response from remote-gw3
```

Una condición de error de expiración ocurre porque las solicitudes se emiten al dispositivo, pero el agente que reside en el dispositivo no entiende el formato `bulk request` SMNP. Como resultado, el agente no responde al paquete y el comando simplemente expira. En la práctica, muchos dispositivos del trabajo en red se han actualizado para soportar la operación `bulk request`.

Por tanto, este problema disminuirá paulatinamente a medida que los viejos dispositivos se remplazan o actualizan.

Nótese que **snmpbulkwalk** exige la utilización de las opciones de seguridad `2c` o `2p`. Si se intenta invocar **snmpbulkwalk** sin especificar cuál versión del protocolo SNMP se usa, se presenta el siguiente error:

```
snmpbulkwalk: Cannot send V1 PDU on V2 session
```

De manera predeterminada, el comando **snmpbulkwalk** buscará todo el árbol MIB-II si no se especifica un objeto o grupo MIB en la línea de comandos. Desde un punto de vista práctico, esto hace que la herramienta sea un poco más fácil de usar, si se desea el almacén completo de los objetos MIB-II.

Herramienta snmpconf

El comando **snmpconf** suministra la habilidad de configurar dispositivos usando el comando **snmpset**, de acuerdo con los objetos MIB definidos en el archivo de configuración. El archivo de configuración puede contener una lista de objetos y valores MIB que se colocarán en un dispositivo SNMP. Esta herramienta suministra un mecanismo automatizado para aplicar información estándar de configuración a varios dispositivos o inclusive, a un solo dispositivo que debe configurarse de la misma manera más de una vez. Considérese, por ejemplo, la instalación de varios dispositivos enrutadores. Sería fácil crear un archivo de configuración que contenga muchos de los objetos comúnmente configurados que comparten los dispositivos enrutadores. Tal configuración podría usarse cuando cada uno de los dispositivos se instale por primera vez. Comparada con la configuración manual de cada enrutador, el uso del archivo de configuración reduce la cantidad de tiempo necesario para instalar un dispositivo, y disminuye la probabilidad de errores de configuración.

El archivo de configuración asociado con **snmpconf** puede contener entradas de objeto MIB, comentarios y líneas en blanco. Se pueden añadir comentarios mediante el uso del signo de número (#). Una entrada de configuración válida consta de un tipo, una OID, un valor y una cadena de comentario. El listado 8.4 muestra un ejemplo del archivo de configuración.

En el listado 8.4, cada una de las tres entradas de configuración contiene el carácter `S` en el primer campo, el cual identifica el tipo de objeto MIB listado en el segundo campo. La `S` significa

que el objeto es de tipo STRING. Este campo se interpreta como una cadena y, por tanto, no debe contener ningún espacio extra. Así, "Virginia Maxwell" es aceptable, pero "Virginia Maxwell" (con el carácter de espacio) no. Además, no se permite el uso de comillas dobles. Si se usa la cadena "Virginia Maxwell", esto dará como resultado un error de configuración. La herramienta **snmpconf** define dos tipos adicionales: I para INTEGER y A para IPADDRESS.

Listado 8.4 Ejemplo del archivo SNMPCONF.

```
#
# Setup standard system group information
#
S .1.3.6.1.2.1.1.4.0 Susan-Maxwell        system contact
S .1.3.6.1.2.1.1.5.0 host.dt.com          system name
S .1.3.6.1.2.1.1.6.0 Graphics_Lab         system location
```

El segundo campo es el objeto MIB que será cambiado al valor contenido en el tercer campo. Nótese que sólo es soportada por **snmpconf** una OID completamente calificada expresada en notación punteada. Cualquier otro formato, como un formato de nombre punteado, por ejemplo,

```
iso.org.dod.internet.mgmt.mib-2.system.sysLocation.0
```

o cadenas MIB abreviadas causará errores de configuración.

Como se expresó antes, el tercer campo es el valor en el cual se establecerá el objeto MIB. Como tal, éste debe corresponder al tipo de carácter especificado en el primer campo. Por tanto, en este ejemplo, cada una de las líneas de configuración espera una cadena en este campo. El último o cuarto campo es para comentarios; suministra un comentario que debería usarse para identificar al objeto y valor MIB. En este caso, los objetos MIB son los objetos contact (sysContact), name (sysName) y location (sysLocation) del grupo system MIB-II.

Una vez el archivo de configuración contenga los objetos y valores MIB necesarios y/o requeridos, puede usarse en un dispositivo SNMP. La sintaxis básica del comando **snmpconf** es la siguiente:

```
snmpconf hostname community configuration_file
```

Así, supóngase que la configuración anterior se almacena en un archivo llamado `config1` y se desea aplicar este archivo al dispositivo llamado `nicodemus`, el cual también tiene una cadena de comunidad `private`. Dada esta información, podría usarse el siguiente comando:

```
# snmpconf nicodemus private config1
```

Ejecutando este comando se presenta la siguiente salida:

```
#
# Setup standard system group information
#
Set display string 'system contact' to Susan Maxwell
Set display string 'system name' to host.dt.com
Set display string 'system location' to Graphics_Lab
Done.
```

Nótese que los comentarios definidos se presentan junto con una confirmación de que cada comando **set** tuvo éxito. Si se quisiera verificar esto, simplemente se ejecuta un **snmpwalk** en el grupo `system` MIB-II. Entonces, el comando

```
# snmpwalk nicodemus public system
```

producirá la siguiente salida:

```
system.sysDescr.0 = unknown
system.sysObjectID.0 = OID:
enterprises.ucdavis.ucdSnmpAgent.linux
system.sysUpTime.0 = Timeticks: (260363) 0:43:23.63
system.sysContact.0 = Susan-Maxwell
system.sysName.0 = host.dt.com
system.sysLocation.0 = Graphics_Lab
system.sysServices.0 = 72
```

El *host* `nicodemus` está corriendo el agente UCD SNMP, y la salida anterior muestra sólo parte del grupo `system`. En este ejemplo se puede confirmar que los objetos `sysContact`, `sysName` y `sysLocation` se han actualizado de acuerdo con el archivo de configuración **snmpconf**.

El software **snmpconf** suministra un par de formas para especificar el archivo de configuración requerido. Primero, el nombre del archivo puede darse en la línea de comandos, como en el ejemplo precedente; **snmpconf** examinará el directorio actual para encontrar el archivo. A continuación, puede suministrarse el archivo con un nombre de ruta completamente calificado. Por

último, si se ha fijado la variable de entorno SNMPCONF_LIB, el software buscará el directorio especificado en esta variable.

La herramienta **snmpconf** suministra algunas verificaciones de error básicas, y cuando ocurren problemas, presenta información útil. Sin embargo, es difícil determinar la causa de ciertos tipos de problemas. Por ejemplo, si se suministra una cadena de comunidad incorrecta o el dispositivo dado no está actualmente en la red, se presentará el mismo mensaje de error. Entonces, el comando

```
# snmpconf didymus private config1
# snmpconf monet public config1
```

produce los mismos errores, como se muestra, pero por diferentes razones:

```
#
# Setup standard system group information
#
UNABLE to set display string 'system contact' to Susan-Maxwell
UNABLE to set display string 'system name' to host.dt.com
UNABLE to set display string 'system location' to
  Graphics_Lab
Done.
3 errors encountered.
```

El primer comando intenta establecer objetos MIB en el sistema didymus, al cual no se puede tener acceso actualmente en la red; es decir, el dispositivo fue apagado cuando se ejecutó este comando. El segundo comando utiliza la cadena de comunidad incorrecta. En este caso, la cadena de comunidad correcta es private no public. Como se puede observar, la herramienta no responde de forma diferente a un sistema apagado y a una cadena de comunidad incorrecta, lo que dificulta para usted determinar la causa de los errores particulares. Sin embargo, si se tiene cuidado, se puede evitar este problema por medio de la emisión del comando **ping** antes de ejecutar el **snmpconf**. De esta manera, se puede evitar este tipo de ambigüedad y abordar los problemas por separados de manera apropiada.

El software **snmpconf** suministra dos opciones de línea de comandos. La primera, **-version**, presenta información de versión para la herramienta **snmpconf** en sí y para la biblioteca de soporte CMU. La segunda, **-quiet**, instruye al **snmpconf** para suspender los mensajes cuando se ejecuta el comando.

Herramienta tkmib

La utilidad `tkmib` suministra una fachada GUI para muchas de las herramientas Linux SNMP, incluidas **snmpget, snmpgetnext, snmpwalk** y **snmptable.** Este software se conoce como un "explorador MIB", el cual puede usarse para interrogar a un agente SNMP respecto a objetos MIB específicos y valores asociados. Para invocar esta herramienta, se emite el comando **tkmib** en un shell UNIX. Cuando el software se inicia, presenta una ventana bastante grande que contiene varios menús, paneles y botones, como se muestra en la figura 8.4.

La ventana contiene cuatro paneles principales: el panel de la vista del árbol MIB, el panel de definición del objeto MIB, el panel de descripción de la MIB y el panel de valor MIB.

Usar el software para consultar un agente es sencillo: se navega hasta la cadena OID deseada que se halla en el panel superior, se introduce un nombre de *host* o dirección IP en el cuadro de texto del *host* objetivo y se pulsa uno de los botones apropiados de operación SNMP.

Las funciones de los botones `stop`, `get`, `getnext` y `walk` debería ser bastante fácil de entender. Desafortunadamente, la función `graph` no parece funcionar de manera correcta; ésta produce un gran número de mensajes de error cuando se invoca. El botón restante, `table`, merece una explicación más amplia. Este botón suministra una gran forma de visualizar las tablas SNMP. Anteriormente se analizaron las funciones del comando **snmptable**, el cual presenta las tablas MIB. Una de las características principales de esta herramienta es la capacidad de dar formato a los datos como entrada para otras aplicaciones y programas. También muestra el formato de las tablas. El botón `table` con `tkmib` se usa principalmente para revisar las tablas MIB, y hace un mejor trabajo que el comando **snmptable** al formatear los elementos de una tabla particular.

Considérese la tabla `tcpConnTable`, donde cada índice se construye con un par dirección IP y puerto. A primera vista, es un poco difícil de entender cómo está organizada esta tabla. Sin embargo, con la ayuda de la herramienta `tkmib`, la tabla es más fácil de entender. La figura 8.5 muestra la vista de `tcpConnTable` que se recuperó de un dispositivo con una dirección IP `10.0.2.201`. Para obtener este listado, se seleccionó la tabla `tcpConnTable` (la cual es del grupo MIB-II `tcp`) y se pulsó el botón `table`.

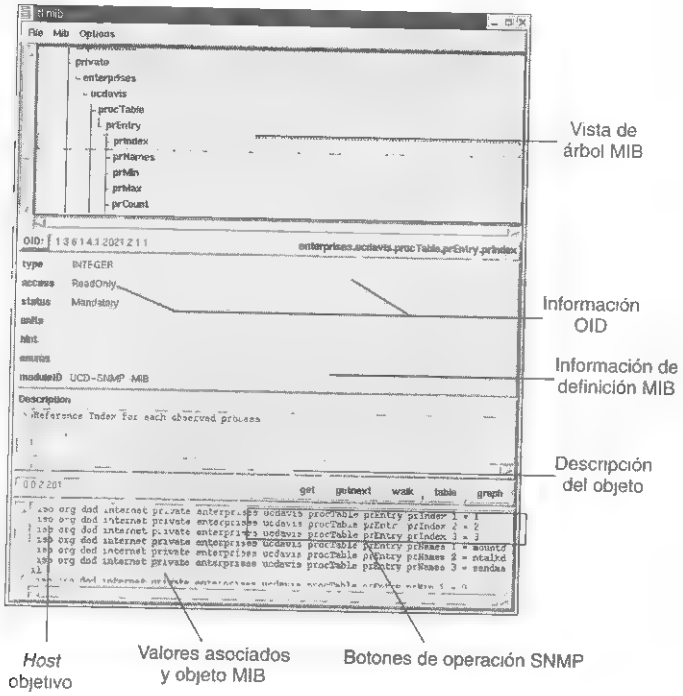


Figura 8.4 Ventana principal **tkmib**.

El panel inferior de la figura contiene una barra de desplazamiento que se usa para listar todas las columnas de la tabla. Esto incluye `tcpConnState`, `tcpConnLocalAdress`, `tcpConnLocalPort`, etc. Nótese, además, la columna `Index`, la cual contiene el índice para cada una de las filas en la tabla.

Elementos del menú

La herramienta `tkmib` suministra tres menús principales: `File`, `Mib` y `Options`. El menú `File` contiene un único elemento, `Quit`, el cual se explica por sí mismo. El siguiente menú, `Mib`, contiene tres elementos: `Find a mib node`, `Load a New Mib File` y `Load a New Mib Module`. El subelemento `Find a mib node` puede buscar las MIB existentes para un objeto o grupo específico. Por ejemplo, suponga que se está buscando el objeto MIB `udpInDatagrams`. Primero, se selecciona el submenú `Find a mib node` del menú `Mib`, para presentar la ventana de diálogo que se muestra en la figura 8.6.

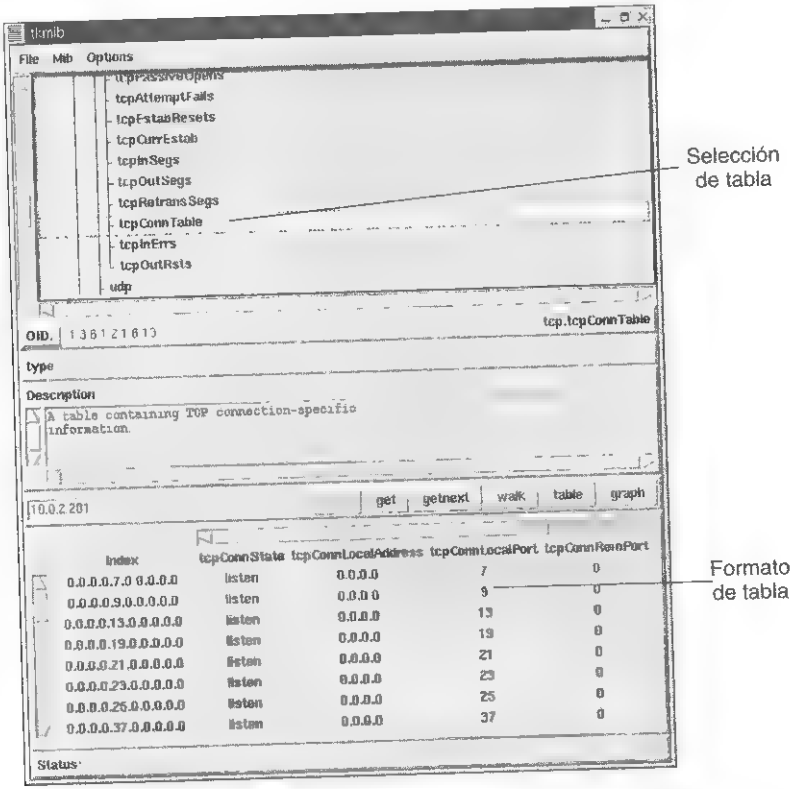


Figura 8.5 Presentación del botón *table*.

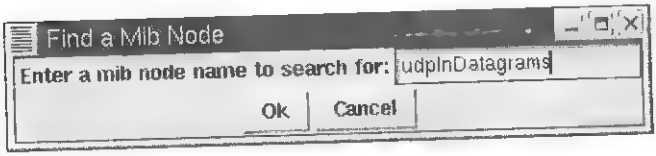


Figura 8.6 Ventana de diálogo de búsqueda de nodo.

Se introduce el nombre de objeto `udpInDatagrams` y se pulsa el botón `OK`. Si se puede encontrar el objeto, el panel del árbol MIB se actualizará para presentar el objeto deseado. De otra manera, se presentará un diálogo de error, indicando que el objeto no se pudo encontrar. Los otros elementos del menú cargan módulos MIB adicionales y archivos en `tkmib`.

Nuevos archivos MIB se pueden importar directamente en la herramienta `tkmib`, de modo que esos objetos OID se presentarán

si se utilizan las cadenas de nombre estándar. Para cargar una nueva MIB, se selecciona el elemento de menú `Load a New Mib File`, y aparecerá una nueva ventana de diálogo, como se muestra en la figura 8.7.

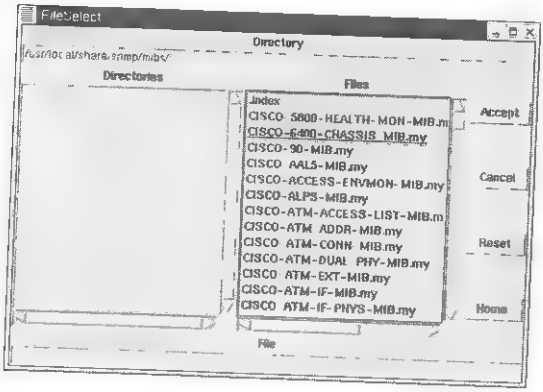


Figura 8.7 Ventana de diálogo de carga del archivo MIB.

Se introduce el nombre en el cuadro de texto de la parte de abajo o se selecciona del panel a la derecha el nombre del archivo que representa la MIB que debería cargarse. En este ejemplo, se importará una MIB Cisco llamada `CISCO-6400-CHASSIS-MIB.my`. Esta MIB contiene elementos utilizados para describir productos Cisco de conmutación y enrutado basados en *chassis*. Una vez se hace una selección, el panel de vista de árbol se redibujará y presentará los elementos del árbol predeterminados. Para establecer si el archivo MIB fue importado como se esperaba, se navega hasta donde debería comenzar el objeto MIB.

En este ejemplo, se creó una serie de grupos relacionados con Cisco desde la rama `enterprises`. La figura 8.8 muestra una vista instantánea del panel de visualización de la copa del árbol que contiene justo la vista MIB que presenta los nuevos grupos MIB definidos bajo la rama Cisco. Si se cargan diferentes MIB específicas de vendedor, los nombres específicos de los vendedores deberían aparecer también en esta lista.

El menú restante contiene cuatro elementos: `Display`, `Time between graph polls`, `Port number` y `Community Name`. El submenú `Display` controla cada uno de los atributos de los objetos MIB que se presentan en el panel de descripción del objeto. El submenú contiene un número de casillas de verificación, una

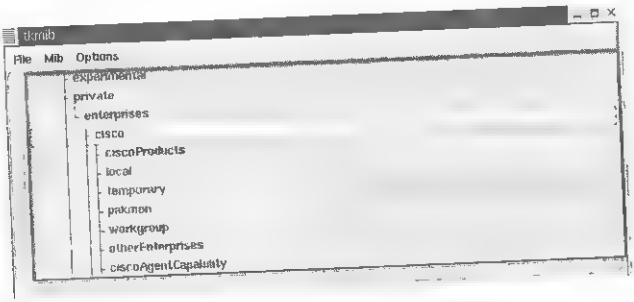


Figura 8.8 Vista de árbol para la rama Cisco.

para cada propiedad MIB. Cuando una de las casillas está sin marcar, el atributo MIB apropiado es removido del panel del objeto.

La opción de menú `Time between graph polls` controla la cantidad de tiempo que transcurrirá por cada recuperación de objetos MIB utilizados en un gráfico. De manera predeterminada, son cinco segundos. El elemento `Port number` del submenú define el puerto TCP utilizado cuando se consulta al agente objetivo SNMP. El predeterminado es el puerto estándar `161`. Para cambiar la predeterminación, se selecciona el elemento `Port number` y cuando se presenta un nuevo cuadro de diálogo (mostrado en la figura 8.9), se introduce el puerto deseado y se presiona el botón `Ok`.

Figura 8.9 Cuadro de diálogo número de puerto.

La contraseña de la cadena de comunidad también puede cambiarse. El valor de `public` se establece de manera predeterminada. Para cambiar la predeterminación, se selecciona el elemento `Community Name` del submenú y se introduce el nuevo valor en la nueva ventana de diálogo que se presenta, como se muestra en la figura 8.10.

Desafortunadamente, los cambios de cualquiera de los elementos en el menú `Options` no se guardan permanentemente

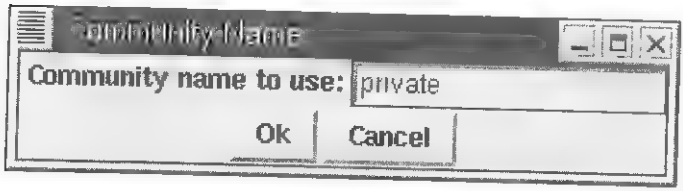


Figura 8.10 Cuadro de diálogo de la cadena de comunidad.

cuando se termina la aplicación. Por tanto, cualquier valor alterado deberá cambiarse cada vez que se inicia la aplicación.

09

Herramientas habilitadas para la Web

Si en el capítulo anterior se analizan las herramientas y utilidades de la administración de red para la línea de comandos UNIX y algunas GUI especializadas, en este capítulo se presentan las herramientas de administración de red que pueden usarse desde un explorador Web. Nadie puede desestimar el impacto significativo que la Web ha tenido en el desarrollo de aplicaciones, interfaces comunes y facilidad de uso, específicamente en el área de la administración de red. Debido a la inmensa popularidad de la computación basada en la Web, es razonable incluir algunas herramientas que se pueden usar a través de exploradores Web comunes.

Estas herramientas incluyen tanto el software basado en el SNMP como el software que no se basa en el SNMP. Las herramientas que no son SNMP se han seleccionado para aquellos sitios de red que todavía tienen que implementar la funcionalidad SNMP o sólo necesitan una herramienta de sistema que se pueda implementar sin mucha dificultad. Para aquellos que ya utilizan el SNMP, las herramientas en esta sección serán una adición bienvenida para cualquier conjunto (suite) de software existente.

Este capítulo analiza las herramientas MRTG y NTOP.

Herramienta MRTG

Descripción general

La herramienta MRTG –graficador de tráfico de múltiples enrutadores– puede hacer seguimiento y graficar el tráfico de la red obtenido de dispositivos de trabajo en red que usan el SNMP. Como el nombre expresa, presenta la utilización del tráfico y otra información estadística recogida de enrutadores y otros dispositivos de red. Ésta genera páginas HTML e imágenes GIF, suministrando una representación visual del rendimiento de la red a través de un explorador Web. El rendimiento de la red es uno de los aspectos más importantes de la administración de red. Esta herramienta facilitará señalar tanto los problemas de rendimiento de los dispositivos como los problemas de rendimiento de la red.

Por ejemplo, dada la topología de la red que se muestra en la figura 9.1, MRTG puede hacer un seguimiento a la red desde el enrutador remoto llamado `remote-gw` y otros dispositivos importantes, como el conmutador/enrutador vertebral y el conmutador de grupo de trabajo. Debido a que esta red es crítica, es importante hacer un seguimiento al tráfico de la red que va hacia el enrutador y sale de éste. Específicamente, es importante hacer un seguimiento a la interfaz serial y a las interfaces Ethernet para asegurarse de que no se sobrecarguen. Además, a la red vertebral y a cualquier subred adjunta debería también hacérsele seguimiento para asegurar operaciones sin contratiempos.

Debido a que MRTG puede hacer seguimiento a cualquier dispositivo que dé soporte al SNMP, puede hacer seguimiento tanto al enrutador lateral como al enrutador vertebral, al igual que a otros dispositivos críticos. Un enrutador lateral (u otro dispositivo) está más cerca de las estaciones de trabajo y de los servidores de los usuarios, mientras que los dispositivos vertebrales son

aquellos localizados en el corazón o centro de la red. Lo importante aquí es que MRTG se puede usar para recolectar información vital desde muchos dispositivos en la red, con el fin de obtener un mejor cuadro de rendimiento.

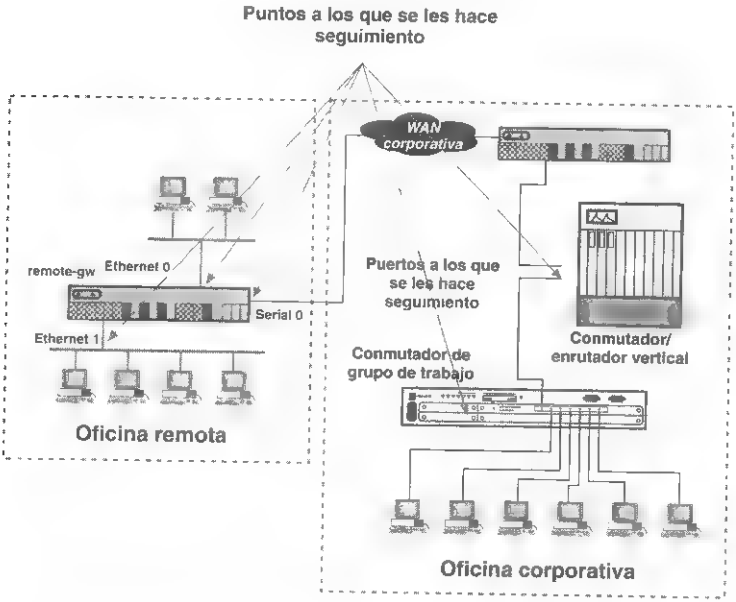


Figura 9.1 Ejemplo de una red de oficina.

Para recolectar información, MRTG utiliza principalmente el comando **snmpget**, el cual se analizó previamente. Sin embargo, se puede personalizar el MRTG para presentar información obtenida a través de otros medios. Por ejemplo, MRTG puede presentar la carga del sistema UNIX y la utilización del espacio del disco. Además, debido a que MRTG puede sondear cualquier objeto MIB SNMP, se pueden recuperar muchas mediciones adicionales de rendimiento y objetos MIB. Como fruto de estas capacidades, existen muchos usos prácticos para el MRTG, incluidos los siguientes:

- presenta la utilización de la red en los enlaces LAN o WAN
- presenta la carga de la CPU del sistema en los dispositivos de red
- presenta la utilización de la memoria del sistema en los dispositivos de red

- presenta el uso del módem en servidores de acceso remoto

Conceptualmente, el MRTG sondea objetos MIB específicos de rendimiento desde los dispositivos y presenta los valores en un formato gráfico usando HTML. La figura 9.2 muestra el flujo de alto nivel del MRTG. La operación básica incluye recolectar datos desde los dispositivos externos y crear archivos asociados de registro para almacenar los datos. Una vez creados los archivos de registro, se pueden generar los reportes HTML. Además, el MRTG debe tener un archivo de configuración que le diga al programa a cuáles dispositivos hacerles seguimiento, y que controle los parámetros específicos operativos y de formato. Desafortunadamente, cada red es diferente, y el usuario debe construir este archivo de configuración. No obstante, MRTG suministra una herramienta automatizada que lleva a cabo gran parte del traba jo necesario para construir una configuración específica para cada red.

Vista general de una página Web

De modo predeterminado, MRTG presenta el resultado de los sondeos de la red en una serie de gráficos, junto con información adicional del dispositivo para cada interfaz a la que se la ha hecho un seguimiento. Suministra un total de cuatro gráficos, que muestran los bytes transmitidos y recibidos por segundo, en promedios diarios, semanales, mensuales y anuales.

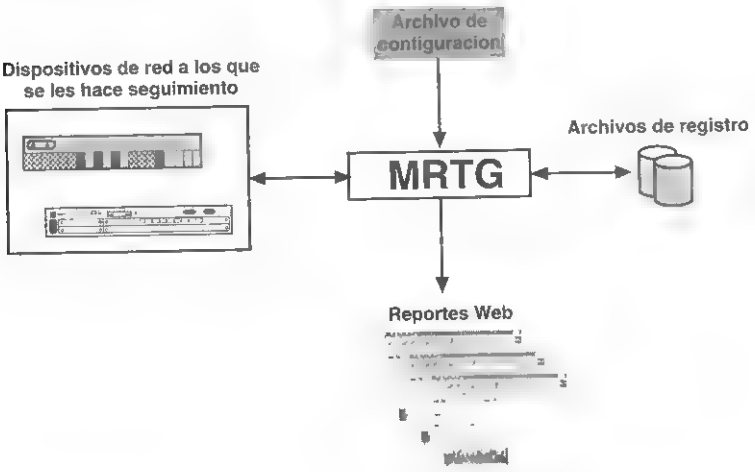


Figura 9.2 Operación básica del MRTG.

Este reporte incluye información obtenida a través del SNMP desde los grupos MIB-II `system` e `interfaces` del dispositivo `remote gw`. Consúltese el capítulo 6, "Vista general de MIB-II", para obtener información adicional respecto a estos grupos MIB. El campo `System` se deriva de los objetos `sysName` y `sysLocation`, mientras el campo `Maintainer` se obtiene del objeto `sysContact`. La cadena `Interface` representa la interfaz física de hardware contenida en el dispositivo, y el número (1) indica que ésta es la primera interfaz. Nótese en la figura 9.3 que la interfaz es etiquetada `Ethernet0`, lo cual significa que es la primera interfaz Ethernet en el dispositivo. Está etiquetada de esta forma porque los enrutadores Cisco siempre numeran sus interfaces arrancando de cero. La figura 9.3 también muestra sólo un reporte parcial MRTG que contiene alguna información de dispositivo además del gráfico de uso diario.

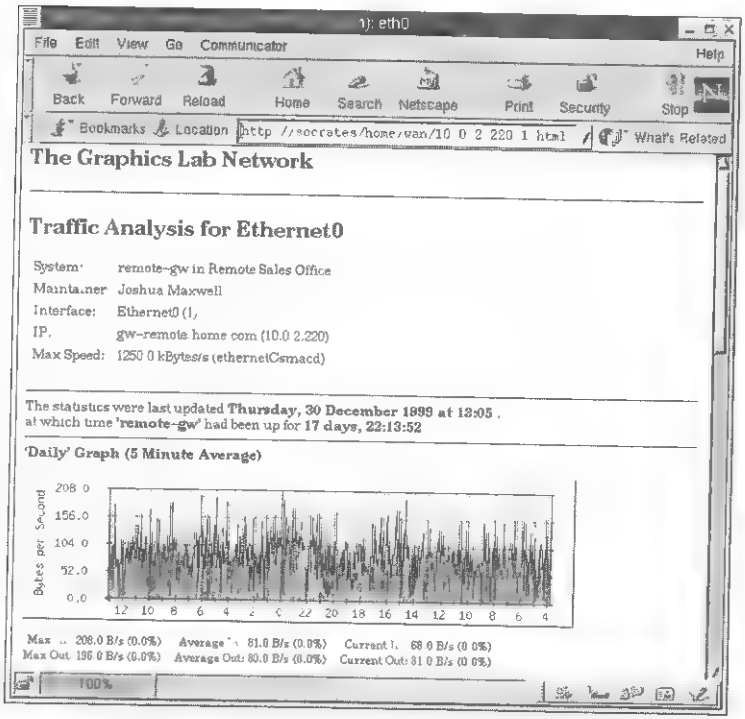


Figura 9.3 Reporte Web MRTG.

La cadena `IP` indica que no se encontró ningún nombre de *host* para la dirección IP de este dispositivo. Esto no es realmente un problema porque algunas veces el acceso a muchos dispositivos de trabajo en red se logra a través de la dirección IP y no del nombre del *host*. Para corregir esto, se introduce el nombre del *host* válido en el archivo local `/etc/hosts` o, si se está corriendo el DNS, se adiciona la entrada en el archivo de configuración del servidor DNS.

El campo `Max Speed` se calcula desde el objeto `IfSpeed` que fue dividido entre ocho. Esto representa el número máximo de bytes por segundo que la interfaz puede soportar teóricamente. El valor de ocho se seleccionó porque hay ocho bits en un solo byte. El campo `Max Speed` también confirma que esta interfaz es Ethernet, presentando el tipo como `ethernetCsmacd`, el cual se deriva del objeto `ifType` contenido en el almacén MIB-2. El reporte también presenta la fecha y hora en que fue actualizado, y por cuánto tiempo ha estado corriendo el dispositivo desde que fue iniciado por última vez o desde que fue reiniciado por última vez el componente de administración de la red del dispositivo.

El gráfico diario muestra bytes por segundo en un periodo aproximado de 36 horas, usando un promedio de cinco minutos. Este promedio es la diferencia entre sondeos sucesivos de bytes transmitidos o recibidos, no de totales acumulativos o conteos simples. En otras palabras, el valor que se presenta es la diferencia entre los valores actuales y previos y se divide entre el tiempo transcurrido entre estas lecturas. El eje horizontal (eje *x*) muestra el tiempo en incrementos de dos horas, mientras el eje vertical (eje *y*) muestra bytes en un rango entre cero y un valor máximo dado. El gráfico presenta bytes transmitidos dentro y fuera de la interfaz, obtenidos de los objetos MIB `ifOutOctets` e `ifInOctets`, respectivamente.

La parte inferior del reporte muestra las leyendas; en este caso, el tráfico que entra se presenta en verde, mientras que el tráfico que sale se presenta en azul. Sin embargo, debido a que este libro es en blanco y negro, el verde de la gráfica aparece en gris y el azul aparece en negro. En la sección llamada "Personalización de los reportes MRTG" se indicará cómo cambiar los colores que se usan en los gráficos. Además, la parte inferior del gráfico contiene un resumen corto que lista los bytes por segundo entrantes y salientes máximos, promedio y actuales. Esto puede ayudar a localizar problemas rápidamente. Usando las opciones de

personalización disponibles con el MRTG, pueden cambiarse estos colores y otros atributos de los reportes y gráficos.

Un reporte MRTG también contiene gráficos adicionales para totales semanales, mensuales y anuales. El gráfico semanal contiene promedios semanales de los siete días anteriores, con el día actual presentado a la izquierda del gráfico, de modo predeterminado. El reporte mensual muestra las últimas cinco semanas, con la semana actual a la izquierda y las semanas previas a la derecha. Los datos en este gráfico se han promediado a partir de un periodo de dos horas. El gráfico anual abarca los doce meses previos, con el mes actual mostrado a la izquierda, y presenta los datos como un promedio diario. La figura 9.4 muestra estos tres gráficos.

Como se puede observar, todos los gráficos crecen desde la izquierda, lo cual está predeterminado. Los gráficos pueden hacerse de tal modo que crezcan desde la derecha, habilitando una de las opciones de comando **MRTG**. Además, de modo predeterminado, las páginas del reporte se actualizan automáticamente cada cinco minutos cuando se obtiene información nueva de cada una de las interfaces a las que se les ha hecho seguimiento. También se puede personalizar este intervalo de actualización (o tasa de refresco).

Configuración básica del MRTG

El MRTG usa un archivo de configuración para especificar a cuáles dispositivos se les debe hacer seguimiento. El paquete MRTG no viene con una configuración predefinida porque los dispositivos listados probablemente no concordarían con los de su sitio. En cambio, el paquete incluye un programa utilitario llamado **cfgmaker**, el cual se usa para construir una configuración básica, de modo que no se tiene que crear una a partir de cero. El propósito de esta utilidad es sondear un dispositivo de red en busca de objetos específicos SNMP (por ejemplo, tabla `interfaces`) y construir un archivo de configuración conveniente para el MRTG. La sintaxis de **cfgmaker** es la siguiente:

```
cfgmaker community@device
```

Se requieren dos argumentos: la cadena de comunidad del dispositivo y el nombre del *host* (o dirección IP). Debido a que **cfgmaker** escribe a la salida estándar, la salida debería ser redirigida hacia un archivo cuando se quiera guardar los resultados. Para ilustrar el uso de esta utilidad, supóngase que se desea

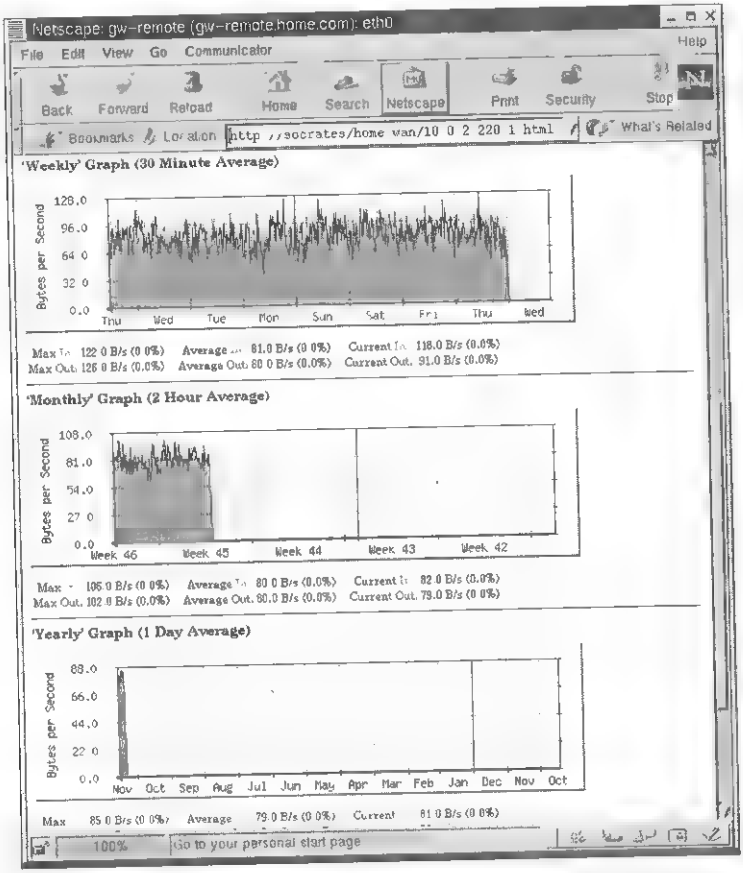


Figura 9.4 Reportes MRTG semanales, mensuales y anuales.

hacer un seguimiento a un solo enrutador en una oficina remota. En este ejemplo, se tiene un enrutador Cisco serie 2500, llamado `remote-gw`, con una cadena de comunidad `louvre`. Por tanto, el comando:

```
# cfgmaker louvre@remote-gw > cisco.conf
```

sondea al dispositivo `remote gw` y construye el archivo de configuración `MRTG` con el nombre `cisco.conf`. Suponiendo que los argumentos suministrados al **cfgmaker** son correctos y que el dispositivo responde a la solicitud SNMP, el listado 9.1 muestra la configuración que generaría (menos los comentarios agregados por el **cfgmaker**) la secuencia de comandos **cfgmaker**:

Listado 9.1 Ejemplo del archivo de configuración **cfgmaker**.

```
Target[10.0.2.75.1]: 1:public@10.0.2.75
MaxBytes[10.0.2.75.1]: 1249994
Title[10.0.2.75.1]: remote gw (No hostname defined for IP
address): Ethernet0
PageTop[10.0.2.75.1]: <H1>Traffic Analysis for Ethernet0
 </H1>
 <TABLE>
   <TR><TD>System:</TD><TD>remote-gw in Graphics Lab 11-
B</TD></TR>
   <TR><TD>Maintainer:</TD><TD>Steve Maxwell</TD></TR>
   <TR><TD>Interface:</TD><TD>Ethernet0 (1)</TD></TR>
   <TR><TD>IP:</TD><TD>No hostname defined for IP address
(161.135.59.129)</TD></TR>
   <TR><TD>Max Speed:</TD>
       <TD>1250.0 kBytes/s (ethernetCsmacd)</TD></TR>
</TABLE>
#-  -  -  -
```

de número (#) seguido de una serie de caracteres " - " (como se muestra en la última línea del listado 9.1). Este enrutador en particular contiene solamente unas pocas interfaces de red, y la palabra clave **target** corresponde a cada interfaz del enrutador. Ésta es una observación clave; cualquier dato al que se haga seguimiento y que sea presentado por el MRTG requiere una entrada **target**.

Este enrutador en particular contiene tres interfaces, una Ethernet y dos serial, pero **cfgmaker** sólo reporta dos interfaces. Debido a que la tercera interfaz está administrativamente abajo, el **cfgmaker** omitió esta interfaz. Ya sea que esta interfaz no se hubiese configurado en el enrutador o inhabilitado por otra razón. En cualquier evento, el **cfgmaker** incluye las primeras dos interfaces pero inserta un carácter de comentario enfrente de la tercera interfaz en el archivo `cisco.conf`, así que no será sondeada con el software MRTG, como se muestra en el listado 9.2.

Listado 9.2 Entrada de interfaz comentada.

```
########
######## This Interface is one of the following
######## - administratively not UP
######## - it is in test mode
########   it is a softwareLoopback interface
######## - has a unrealistic speed setting
######## It is commented out for this reason.
########
#
# Target[10.0.2.75.3]: 3:public@10.0.2.75
# MaxBytes[10.0.2.75.3]: 193000
# Title[10.0.2.75.3]: remote-gw (No hostname defined for IP
    address): Serial1
# PageTop[10.0.2.75.3]: <H1>Traffic Analysis for Serial1
#  <BR></H1>
#  <TABLE>
#    <TR><TD>System:</TD><TD>remote-gw in </TD></TR>
#    <TR><TD>Maintainer:</TD><TD></TD></TR>
#    <TR><TD>Interface:</TD><TD>Serial1 (3)</TD></TR>
#    <TR><TD>IP:</TD><TD> ()</TD></TR>
#    <TR><TD>Max Speed:</TD>
#        <TD>193.0 kBytes/s (propPointToPointSerial)</TD></TR>
# </TABLE>
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

Inhabilitar la interfaz es una buena idea porque no es razonable tener al MRTG sondeando interfaces no activas, cuando no hay tráfico, produciendo así gráficos vacíos. Además, si la

interfaz se habilita y se pone en servicio en una fecha futura, se trata sólo de quitar los comentarios en la entrada de la interfaz, permitiendo así al MRTG hacerle un seguimiento al puerto. El archivo de configuración del MRTG utiliza el lenguaje de formato HTML y, en particular, construye una tabla para dar formato a la información del dispositivo, como se denota por las etiquetas `<TABLE>` y `<TABLE>`. La configuración también incluye una o más palabras clave que controlan a qué dispositivos se les hace seguimiento y cómo se grafican los datos. Cada palabra clave contiene la siguiente sintaxis:

```
Command [label] args
```

donde **Command** es uno de los comandos que se muestran a continuación, seguido de una etiqueta encerrada entre corchetes y uno o más argumentos u opciones de comando.

Cada elemento o interfaz al que se le ha hecho seguimiento en el archivo de configuración normalmente contiene las siguientes palabras clave:

- **Target**
- **MaxBytes**
- **Title**
- **PageTop**

La palabra clave **target** indica al MRTG a qué debería hacerle seguimiento. El formato básico es:

```
Target[label]: port:community@device
```

La palabra `label` se utiliza como un identificador, de modo que el MRTG pueda presentar un grupo de palabras clave en una página HTML. `label` es una combinación de la dirección IP del dispositivo y el número de interfaz. La palabra `port` es el número de interfaz como está definido en el dispositivo `device`, que el MRTG debería sondear con la cadena de comunidad SNMP `community`. El puerto o número de interfaz debe ser una interfaz válida en el dispositivo que se está sondeado. Se pueden determinar las interfaces válidas en un dispositivo dado, con el programa cfgmaker MRTG. Mediante el uso de este programa, todas las interfaces definidas en un dispositivo dado se descubrirán automáticamente. Como otra alternativa, usando un explorador MIB, se puede examinar con detenimiento el árbol MIB del

dispositivo y la tabla de la interfaz, para descubrir las interfaces de modo manual. Sin embargo, si ya se sabe a cuáles interfaces se quiere hacerles un seguimiento, especificarlas manualmente puede ser el método más simple. Usando el ejemplo de configuración anterior, considérese el siguiente comando **Target:**

```
Target[10.0.2.75.1]: 1:public@10.0.2.75
```

Este comando definirá la etiqueta `10.0.2.75.1` e instruirá al MRTG para que haga seguimiento al tráfico desde la primera interfaz del dispositivo con la dirección IP `10.0.2.75` usando la cadena de comunidad `public`. De modo predeterminado, el MRTG sondea los objetos MIB `ifInOctets` e `ifOutOctets` desde un dispositivo especificado y grafica estos datos en el transcurso del tiempo.

La palabra clave **Maxbytes** define el límite superior para cualquiera de las estadísticas de tráfico a las que se les ha hecho seguimiento, y determina el rango del eje y para gráficos que no están a escala. Para cualquier interfaz, la palabra clave **Maxbytes** se especifica normalmente en bytes por segundo, los cuales pueden calcularse dividiendo el ancho de banda máximo de la interfaz entre ocho. Por ejemplo, como la Ethernet 10M tiene un ancho de banda máximo de 10 Mbps, dividiendo esto entre ocho (8), se obtiene `1250000` (10,000,000 / 8 = 1,250,000). Establecer este valor correctamente es muy importante, porque se asegurará que los gráficos presenten información realista. Por ejemplo, en el listado 9.2, el comando

```
MaxBytes[10.0.2.75.1]: 1250000
```

define el máximo de 1,250,000 bytes para el dispositivo asociado con la etiqueta `10.0.2.75.1`. Recuérdese que el programa **cfgmaker** asigna el valor **MaxBytes**, y la razón para seleccionar este valor es que la interfaz es de tipo Ethernet. Tener el valor en bytes por segundo a mano es importante. Por esto, la tabla 9.1 incluye los tipos de interfaz de trabajo en red más comunes. Esta información se ha incluido aquí porque el MRTG no incluye el máximo de bytes por segundo para los gráficos; esto asegura que sean legibles cuando se grafican cargas pequeñas de tráfico.

Tabla 9.1 Bytes por segundo para interfaces diferentes.

Tipo de red	Bytes por segundo
ATM-OC3 (155Mbps)	19,375,000
Gigabit Ethernet (1000Mbps)	125,000,000
Fast Ethernet (100Mbps)	12,500,000
FDDI (100Mbps)	12,500,000
T3/DS-3 (45Mbps)	5,593,250
Ethernet (10Mbps)	1,250,000
Token Ring (16Mbps)	2,000,000
Token Ring (4Mbps)	500,000
T1/DS-1 (1.54Mbps)	193,000
56K	7,000

El MRTG también usa el valor `MaxBytes` para detectar respuestas no válidas cuando sondea dispositivos para estadísticas de tráfico de interfaz.

La palabra clave **Title** etiqueta la página de reporte e incluye usualmente alguna información respecto a la interfaz particular a la que se le está haciendo seguimiento. El ejemplo

```
Title[10.0.2.75.1]: remote-gw (No hostname defined for IP
  address): Ethernet0
```

define el título que está asociado con el sistema `10.0.2.75`. Éste contiene el nombre del *host*, más la interfaz (es decir, `Ethernet0`).

La última palabra clave, **PageTop**, incluye información adicional que se colocará en la parte superior del reporte. Esto permite poner en el reporte más descripción de la interfaz, u otra información.

```
PageTop[10.0.2.75.1]: <H1>Traffic Analysis for Ethernet0
  </H1>
```

En este ejemplo, la cadena `Traffic Analysis for Ethernet0` se incluye cuando se ejecuta la secuencia de comandos **cfgmaker**, y además describe qué representa el reporte. Los marcadores

<H1> y </H1> son etiquetas de encabezamiento HTML. El explorador puede escoger presentar la cadena entre estos marcadores en uno de muchos formatos diferentes, incluidos centrado, texto en negrita, fuente grande, itálicas, color o subrayado. A esta palabra clave se pueden agregar líneas múltiples y, debido a que el HTML normalmente remueve caracteres de nueva línea, la secuencia \n debe usarse si se requieren nuevas líneas.

Uso del MRTG

Después de preparar un archivo de configuración básica para los dispositivos, es tiempo de comenzar a usar el MRTG para hacer seguimiento al rendimiento y producir reportes en estos dispositivos. El comando que sondea los dispositivos para establecer estadísticas de rendimiento y construir las páginas HTML se conoce como **mrtg.** Sin embargo, antes de poder ejecutar este comando, se debe incluir la entrada del comando de la palabra clave **WorkDir** en el archivo de configuración. Esta palabra clave le indica al MRTG dónde crear los archivos de registro y las páginas HTML. La sintaxis de este comando es:

```
WorkDir: path
```

donde `path` es el nombre absoluto de ruta del directorio objetivo que contendrá todo los archivos que se necesitan. Después de adicionar este comando al archivo de configuración, es tiempo de ejecutar el **mrtg** con la nueva configuración, para asegurarse de que no se ha introducido ningún error de sintaxis u otros problemas. Éste es un paso importante porque ayuda a asegurarse de que se capta cualquier problema antes de comenzar a ejecutar el **mrtg** sobre una base más o menos permanente. Suponiendo que el archivo de configuración que se está usando se denomina `cisco.conf`, el comando

```
# /usr/bin/mrtg cisco.conf
```

ejecuta el **mrtg** con este archivo e inmediatamente intenta recolectar datos SNMP de los dispositivo(s) listado(s). Después de que el comando **mrtg** termina crea una serie de archivos en el directorio especificado por el comando **WorkDir.** Supóngase, por ejemplo, que el archivo `cisco.conf` contiene las siguientes entradas de ejemplo:

```
WorkDir: /usr/mrtg/www
Target[10.0.2.75.1]: 1:public@10.0.2.75
```

Los siguientes archivos se crearán en el directorio `/usr/mrtg/www`:

```
10.0.2.75.1 day.gif
10.0.2.75.1-month.gif
10.0.2.75.1-week.gif
10.0.2.75.1-year.gif
10.0.2.75.1.html
10.0.2.75.1.log
10.0.2.75.1.old
```

Todos estos archivos tienen el prefijo de la dirección IP y el número de interfaz, y representan el dispositivo y el puerto a los que el MRTG está haciendo seguimiento. El archivo `10.0.2.71.1.html` contiene el reporte de alto nivel en HTML, que se mostró en la figura 9.1. Cuando se vean los reportes MRTG, se usa este archivo y un explorador estándar Web para presentar los gráficos de rendimiento de la interfaz. Los archivos con la extensión `gif` son los cuatro gráficos que muestran el uso diario, semanal, mensual y anual en bytes por segundo del dispositivo que utiliza el archivo de reporte `10.0.2.75.1`.**html**. Los archivos `10.0.2.75.1.log` y `10.0.2.75.1.old` contienen los datos sondeados SNMP empleados para construir los gráficos.

Para cada interfaz a la que se le hace seguimiento mediante el MRTG, estos siete archivos se crearán automáticamente. Otro aspecto positivo del MRTG es que los reportes y archivos de datos no ocupan una gran cantidad de espacio del disco. De hecho, el MRTG usa un método único de almacenamiento de datos para minimizar la cantidad de espacio utilizado en los gráficos históricos.

Para asegurarse de que el MRTG captura información consistente y completa de los dispositivos a los que se les hace seguimiento, el programa MRTG debe correrse en intervalos regulares. Una de las mejores formas de conseguir esto es a través del uso de la facilidad `cron`. Como el gráfico diario, el cual está disponible en el reporte MRTG, se promedia a partir de un periodo de cinco minutos, es razonable que el **mrtg** se corra cada cinco minutos. Un ejemplo de la entrada **crontab** que haría exactamente esto se muestra enseguida:

```
0,5,10,15,20,25,30,35,40,45,50,55 * * * * /usr/bin/mrtg \
/usr/mrtg/www mrtg.conf
```

Esta entrada ejecutará el programa **mrtg** cada cinco minutos cada día de la semana. El * es un comodín que significa *todo* para las cuatro especificaciones de tiempos siguiendo el campo minutos (`minutes`). Esta entrada se copia en el archivo contrab de root para que corra de manera continua.

Personalización de los reportes MRTG

Se puede personalizar el paquete MRTG para producir reportes y gráficos formateados de forma diferente y para controlar ciertos aspectos operativos del MRTG. Para llevar a cabo esto, se usan etiquetas (o palabras clave) de configuración de objetivo opcionales en el archivo de configuración MRTG. Las palabras clave se pueden dividir en dos grupos: *comandos opcionales globales* y *comandos opcionales objetivo*. Los comandos globales afectan la operación o el comportamiento del MRTG como un todo, mientras que los comandos opcionales objetivo se aplican específicamente a objetivos individuales o reportes de dispositivos y a gráficos.

Comandos globales

La tabla 9.2 lista los cuatro comandos globales soportados por el MRTG. Estos comandos opcionales se colocan en el archivo de configuración que se usa con el programa **mrtg**.

Tabla 9.2 Comandos globales opcionales.

Comando	Descripción
`Refresh`	Especifica con qué frecuencia (en segundos) se debe actualizar el reporte `MRTG`
`Interval`	Especifica qué tan seguido recibe un llamado **mrtg**
`WriteExpires`	Si se establece en `YES`, generará meta-archivos que controlan las etiquetas de expiración para los reportes `MRTG`
`IconDir`	Le dice al `MRTG` que busque en una ubicación alternativa imágenes GIF

El comando **Refresh** especifica qué tan seguido, en segundos, debería recargar el explorador la página del reporte MRTG. El valor predeterminado es 300 segundos (cinco minutos). Ésta es una opción muy útil cuando es importante controlar de una forma precisa la granularidad de la actualización de los reportes MRTG. Por ejemplo, si es necesario tener información del rendimiento casi en tiempo real en ciertos dispositivos críticos, disminuir a **Refresh** hasta 60 segundos puede ser apropiado. Por otra parte, **Refresh** debería ser más grande para aquellos dispositivos menos críticos y que no requieren reportar la información más reciente. Para reducir la tasa de actualización del explorador a un minuto, se coloca el siguiente comando en el archivo de configuración MRTG:

```
Refresh: 60
```

Para que el valor de actualización sea efectivo, debería estar muy cercano al intervalo en el cual el **mrtg** se ejecuta cuando se sondean los dispositivos. La entrada `crontab`, previamente especificada en esta sección, corre al **mrtg** cada cinco minutos. Si el valor de actualización se disminuye de su valor predeterminado, la entrada `crontab` también debería actualizarse con el mismo valor. Considérese la situación en que el valor de actualización es un minuto, mientras que la ejecución del **mrtg** es cada cinco minutos. En este escenario no tendría sentido actualizar los reportes MRTG, porque los datos sólo serían actualizados cada cinco minutos por el **mrtg.** Claramente se nota que el incremento de la granularidad de los reportes MRTG exige que el programa **mrtg** se ejecute más seguido para recolectar puntos adicionales de datos del dispositivo al que se le está haciendo seguimiento.

El comando **Interval** especifica qué tan seguido se ejecuta `MRTG`. El propósito de este comando es generar en el reporte MRTG un metaencabezado que controla el tiempo de vida del reporte con propósitos de caché. Además, el metaencabezado documenta las páginas HTML con el intervalo correcto de llamado. Si el explorador de Web que se está usando requiere de la generación de metaarchivos que controlan la expiración de los reportes MRTG, se usa el comando **WriteExpires**. Estos metaarchivos contienen etiquetas que especifican la expiración tanto de los archivos HTML como de los GIF. Habilitar el **WriteExpires** hará que el MRTG cree archivos adicionales en el directorio de trabajo.

El comando **IconDir** puede especificar un directorio alternativo (en lugar del directorio de trabajo) para contener iconos usados

por el MRTG. Es útil separar los archivos de reporte del dispositivo de las imágenes GIF utilizadas con los reportes. Por tanto, el uso del directorio `graphics` para mantener todas las imágenes, se logra mediante el siguiente comando:

```
IconDir: /graphics/
```

Nótese que el nombre del directorio debe incluir el carácter "/" al final.

Comandos opcionales objetivo

Muchas opciones controlan el formato de los gráficos, pero sólo algunas afectan la forma de presentación del reporte. Todas las palabras clave descritas aquí tienen la misma sintaxis que los otros comandos MRTG, como **Target** y **MaxBytes.** La tabla 9.3 lista estas palabras clave de personalización, y a continuación se analiza cada una.

Tabla 9.3 Opciones de personalización MRTG.

Opción	Descripción
`AddHead`	Como la palabra clave **PageTop**, ésta incluye la entrada especificada en el reporte MRTG. Ésta es una gran forma de agregar información de un sitio de la red específico a los reportes MRTG
`AbsMax`	Especifica el máximo valor que se alcanzará siempre que se grafican los datos
`Background`	Controla el color del fondo de los reportes MRTG
`Colours`	Reemplaza el esquema de color predeterminado
`Directory`	Coloca todos los archivos relacionados con una etiqueta dada en el directorio especificado
`Legend`	Controla las cadenas para la leyenda de color
`Options`	Habilita el uso de conmutadores booleanos adicionales, los cuales controlan además la presentación de atributos de gráficos

Tabla 9.3 Opciones de personalización MRTG. (*continuación*)

Opción	Descripción
`ShortLegend`	Reemplaza la cadena de unidades que, de modo predeterminado, es `b/s`
`Step`	Cambia el paso predeterminado o la extemporaneidad desde cinco minutos (`5 * 60`)
`Suppress`	Inhabilita la presentación de los gráficos diarios, semanales, mensuales o anuales en un reporte MRTG
`Timezone`	Controla zona horaria que se utiliza. Se soportan nombres estándar, como `Japan`
`Unscaled`	Apaga la escala vertical de datos en gráficos de un reporte MRTG
`Weekformat`	Controla cuál función de la API UNIX se usa para dar formato a la información semanal presentada en los gráficos mensuales
`WithPeak`	Habilita la presentación de valores máximos en gráficos del reporte MRTG
`XSize/YSize`	Cambia los tamaños verticales y horizontales de los gráficos de un reporte MRTG
`XZoom/YZoom`	Incrementa el número de pixeles usados con un gráfico
`XScale/YScale`	Gradúa tanto el eje *x* como el eje *y* del gráfico
`Ylegend`	Reemplaza el texto presentado para el eje *y* en el gráfico

La palabra clave **AddHead** agrega un código HTML entre las etiquetas `</TITLE>` y `</HEAD>` HTML en el HTML resultante. Ésta es una buena forma de agregar cualquier información específica del sitio de la red a los reportes MRTG. Por ejemplo, para

agregar la capacidad de enviar correo electrónico individual desde el reporte MRTG, se añade lo siguiente:

```
AddHead[10.0.2.75.1]: <A HREF="mailto:support@monet">Open
  Trouble Ticket</A>
```

Nótese que esta palabra clave requiere una etiqueta y se usa para hacer concordar el **AddHead** con la entrada de comando **Target** existente.

En forma predeterminada, el MRTG gradúa el eje vertical para asegurar que los datos mucho más pequeños que el `MaxBytes` se presenten de tal forma que sean visibles en el gráfico. Si se quiere inhabilitar la función de graduación, se usa la palabra clave **Unscaled**. Este comando acepta un único argumento de cadena que consta de la primera letra del nombre del gráfico que no debería ser ajustado. La cadena incluye `d` para diario, `w` para semanal, `m` para mensual y `y` para anual. Por tanto, el comando

```
Unscaled[10.0.2.71.1]: dw
```

inhabilita la graduación para los gráficos diarios y semanales.

El comando **WithPeak** instruye al MRTG para presentar tasas pico de cinco minutos para tráfico entrante y saliente en interfaces de dispositivo. Normalmente, los gráficos sólo contienen los valores promedio de las variables a las que se les hace seguimiento. Este comando toma una cadena sencilla que contiene la primera letra de los nombres de los gráficos que deberían presentar los valores máximos. La cadena incluye `w` para semanal, `m` para mensual e `y` para anual. En el siguiente ejemplo, tanto el gráfico semanal como el anual contendrán información de picos:

```
WithPeak[10.0.2.71.1]: dw
```

La razón para que el gráfico diario no esté incluido en esta lista es que este gráfico ya muestra información de picos.

En general, el MRTG presenta todos los cuatro gráficos en el reporte de la interfaz. Se puede usar la palabra clave **Suppress** para remover uno o más de estos gráficos del reporte. Esta palabra clave acepta el mismo argumento tanto del comando **Unscaled** como del comando **WithPeak**. Para bloquear la presentación de los gráficos semanales, mensuales y anuales, se usa el siguiente comando:

```
Suppress[10.0.2./1.1]: wmy
```

Para disponer de una forma de organizar mejor los reportes MRTG y los archivos de datos, se suministra el comando **Directory**. En forma predeterminada, el MRTG guarda todos los archivos en el directorio especificado con la palabra clave **WorkDir**. El comando **Directory** acepta un argumento de cadena que representa el nombre de un directorio en el cual se guardan todos los archivos asociados con la etiqueta. Los comandos

```
WorkDir: /usr/mrtg/www/devices
Directory[10.0.2.71.1] gw-1
```

instruyen al MRTG para que guarde todos los archivos asociados con la etiqueta `10.0.2.71.1` en el directorio llamado `gw 1`. Como se puede observar, esto puede ser útil para administrar una gran instalación de dispositivos e interfaces a los que se les hace seguimiento, porque cada interfaz se puede guardar en su propia ubicación. El directorio especificado con el comando **Directory** debe existir antes de correr el MRTG, porque el MRTG no lo creará. Si el MRTG se corre antes de que este directorio exista, producirá un mensaje de error y se malogrará.

La palabra clave **options** suministra otras capacidades para personalizar la forma en que se presentan los gráficos MRTG. Este comando acepta varios conmutadores booleanos, listados en la tabla 9.4, que alteran atributos para controlar varias opciones de graficado.

Tabla 9.4 Opciones de conmutadores booleanos.

Opciones	Descripción
`growright`	Hace que las gráficas crezcan del lado derecho del gráfico
`bits`	Cambia los bits por bytes
`noinfo`	Suprime información de tiempo arriba (*timeup*) y del dispositivo
`nopercent`	Suprime el uso de porcentajes
`gauge`	Trata los valores sondeados como absolutos, en lugar de contadores que se incrementan
`absolute`	Se usa con valores `gauge` para indicar valores que no se incrementan, que se reinician cuando se obtienen los datos

La opción `growright` hará que los datos se agreguen al lado derecho del gráfico, en lugar del izquierdo, lo cual es el comportamiento predeterminado. Ésta también revierte el valor del eje *x* para todos los gráficos.

Con el `MRTG`, todos los valores a los que se les hace seguimiento se presentan en bytes, de manera predeterminada. Con la opción `bits`, estos valores se pueden presentar en bits, en su lugar. Esto tiene un profundo impacto en los gráficos, ya que los valores `bits` parecen producir un mejor gráfico que el gráfico de bytes. Las palabras clave **noinfo** y **nopercent** suprimen información del reporte MRTG. En el caso de **noinfo**, ninguna información de tiempo arriba o adicional del dispositivo se presentará. La palabra clave **nopercent** no mostrará la información de porcentaje en los gráficos.

El conmutador `gauge` trata los valores recolectados como cantidades absolutas en lugar de considerarlos como incrementos a los valores presentados en sondeos previos. Esto es útil para hacer un seguimiento de la información relacionada con el espacio del disco, con la utilización del procesador y con el número de usuarios en el sistema. El ejemplo de seguimiento del espacio del disco analizado en la sección "Seguimiento del espacio del disco" usa la opción `gauge`. El conmutador `absolute` se usa para fuentes de datos que reinician sus valores cuando se obtienen los datos. Esto indica que el MRTG no necesita calcular la diferencia entre el valor presente y el valor previo de sondeo. El valor `absolute` se divide entre el tiempo transcurrido entre los dos últimos sondeos, lo cual hace una gran diferencia respecto de los valores `gauge` analizados previamente.

Nombres objetivos especiales

El `MRTG` suministra tres nombres objetivos especiales que pueden reducir la cantidad de digitación en el archivo de configuración del MRTG. Estos caracteres especiales incluyen el acento circunflejo (^), el signo de dólares ($) y el subrayado (_). El texto de cada palabra clave que usa ^ se agrega al comienzo de cada objetivo subsecuente que se utiliza. Por ejemplo, la secuencia MRTG

```
PageTop[^]: <H1>The Graphics Lab Network</H1>
```

se adicionará a cada comando de palabra clave **PageTop** adicional que se encuentra en el archivo de configuración. Por tanto, dado

```
PageTop[10.0.2.75.1]: <H1>Traffic Analysis for Ethernet0</H1>
```

la salida en el reporte será

```
The Graphics Lab Network
Traffic Analysis for Ethernet0
```

Siempre que el MRTG encuentra una entrada **target**, adiciona la cadena `The Graphics Lab Network` al comienzo de cada comando **target**.

Cuando se usa el carácter **$** en lugar del **^**, la cadena que sigue se agrega a cualquier palabra clave adicional. Por tanto, dado

```
PageTop[$]: <H1> <A HREF="mailto:lisa_ellen@think-tech.
com">Lisa Ellen/A></H1>
```

el enlace anterior de correo electrónico aparece al final de cualquiera de las entradas **PageTop**. Usando el ejemplo anterior, la salida en el reporte será:

```
Traffic Analysis for Etnernet0
Lisa Ellen
```

Si se quiere remplazar un valor existente agregado, simplemente se refina:

```
PageTop[^]: <H1>The Graphics Lab Network</H1>
PageTop[10.0.2.75.1]: <H1>Traffic Analysis for Ethernet0</H1>
PageTop[^]: <H1>The Engineering Lab</H1>
PageTop[10.0.2.100.1]: <H1>Traffic Analysis for ATM1</H1>
```

En esta salida, la segunda instancia de **PageTop[^]** sobreescribe la primera y se usará a lo largo de todo el archivo de configuración, a menos que se encuentren palabras clave adicionales **PageTop[^]**.

Además, se pueden restablecer valores previamente definidos con **^** y **$** a nulo o a no colocados (*unset*), simplemente mediante el uso de caracteres sin ninguna definición de cadena. Por tanto,

```
PageTop[^]:
PageTop[$]:
```

reestablece las palabras clave al estado predefinido de no valor.

El carácter **_** se usa con palabras clave MRTG y especifica un valor predefinido para esa palabra clave. En ausencia de un valor explícito de palabra clave o de un valor agregado, se usará el predeterminado, como en el siguiente ejemplo:

```
Target[10.0.2.75.1]: 1:public@10.0.2.75
```

La definición **Target** con la etiqueta `10.0.2./5.1` puede replicarse sin la necesidad de usar la cadena de etiqueta. Así,

```
MaxBytes[_]: 1249991
Title[_]: remote-gw (): Ethernet0
Options[_]: growright
PageTop[_]: <H1>Traffic Analysis for Ethernet0</H1>
```

son todas entradas MRTG válidas, porque el subrayado representa la etiqueta `10.0.2.75.1`.

Para reiniciar el subrayado, se especifica un valor vacío:

```
Options[_]:
```

Índice maestro

El MRTG puede considerarse como de *interfaz céntrica*. Esto significa que el foco principal está en la interfaz de los dispositivos a los que se les está haciendo seguimiento. Si el MRTG se usa para hacer seguimiento al enrutador Cisco con cinco interfaces, se crearán cinco reportes MRTG por separado. Desafortunadamente, aunque esta estructura es suficiente para un pequeño número de interfaces, llega a ser difícil de usar y administrar cuando se enfrenta con un gran número de interfaces o la combinación de muchos dispositivos e interfaces. Para ayudar en esta tarea, se suministró en la distribución MRTG una secuencia de comandos llamada `indexmaker`, la cual produce un índice de todos los dispositivos e interfaces a los que se les hace seguimiento. Esta secuencia de comandos puede encontrarse en el directorio `/usr/lib/mrtg`. La sintaxis de `indexmaker` es:

```
indexmaker label config result_file
```

Este comando soporta varios argumentos de línea de comandos. La tabla 9.5 muestra muchos de los más importantes. Como mínimo, se requieren tres argumentos: el archivo de configuración MRTG, la etiqueta del gráfico y la expresión regular, que incluye todas las interfaces a las que se les ha hecho seguimiento y que deberían agregarse al índice. Debido a que **indexmaker** escribe a la salida estándar, la salida debe ser redirigida hacia un archivo. Por tanto, el ejemplo de comando

```
# indexmaker -t 'Summary of Interfaces' *.conf > backbone.html
```

construye un índice maestro de interfaz para todas las interfaces contenidas en el dispositivo `10.0.2.75` usando el archivo `backbone.html`.

Tabla 9.5 Argumentos de línea de comandos **indexmaker.**

Opción	Descripción
`-1`	Construye una página Web de una columna. Los gráficos se apilarán uno encima del otro
`-2`	Construye una página Web de dos columnas. Los gráficos se colocan uno al lado del otro, en lugar de uno encima del otro. Éste es el modo predeterminado
`D`	Establece el color de fondo. El predeterminado es blanco (o sin color). El valor debe especificarse en formato hexadecimal. Por ejemplo, `#ffffff` es el código para negro
`-r`	Utiliza la expresión regular especificada para hacer coincidir interfaces objetivo. La predeterminada es "."
`-t`	Establece el `título` de la página del índice. El valor predeterminado es `All targets`
`P`	Establece el análisis de las etiquetas **PageTop** para el nombre del gráfico en la página índice. Éste es el predeterminado
`-p`	No hace el análisis de las etiquetas **PageTop** para el nombre del gráfico en la página índice
`-H`	Establece el uso de etiquetas `<HR>` entre gráficos. Éste es el predeterminado
`-h`	No establece el uso de las etiquetas `<HR>` entre gráficos
`-S`	Ordena los gráficos alfabéticamente. Éste es el tipo de orden predeterminado
`-s`	No ordena los gráficos. Aparecen todos en el mismo orden en que se encontraron en el archivo de configuración

Tabla 9.5 Argumentos de línea de comandos **indexmaker.** (*continuación*)

Opción	Descripción
-N	Agrega un número de secuencia al comienzo de cada gráfico
-n	No numera los gráficos. Éste es el predeterminado
-X	Especifica el ancho para cada gráfico. El valor predeterminado es 500
-Y	Especifica la altura de cada gráfico. El valor predeterminado es 135
-A	Incluye argumentos como comentarios en el código HTML
-a	No incluye argumentos como comentarios
-G	Adiciona gráficos a las páginas de índice. Éste se habilita de modo predeterminado
-g	No adiciona gráficos a las páginas de índice

El programa indexmaker se puede encontrar en el directorio `/usr/lib/mrtg`. El nuevo archivo, cuando se presenta, debería verse similar a la ventana que se presenta en la figura 9.5. Muestra el gráfico diario para cada interfaz y contiene un enlace hacia cada una de las páginas definidas, las cuales contienen los gráficos restantes para esa interfaz.

Información adicional de seguimiento

El MRTG se puede usar para hacer seguimiento a otra información –incluidos objetos adicionales SNMP e información no SNMP–, utilizando el comando **target**. El comando **target** acepta formatos adicionales de entrada, los cuales se pueden usar para hacer seguimiento y graficar la información. Estos formatos incluyen cadenas directas OID, objetos abreviados SNMP y comandos personalizados. El formato directo OID utiliza cadenas de identificador de objeto (OID), expresadas en notación punteada decimal completamente calificada, que especifica cuál objeto debería ser monitoreado por el MRTG. *Completamente calificada*

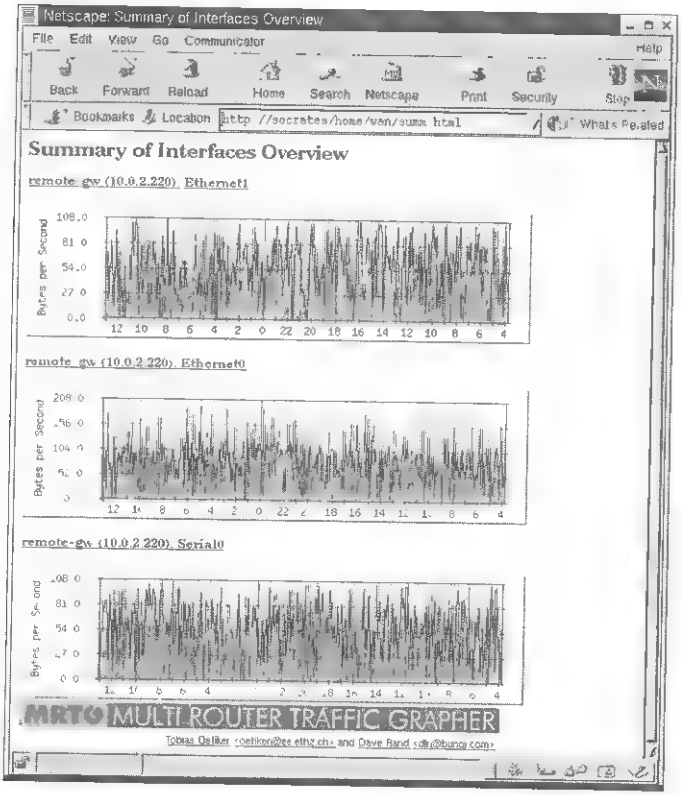


Figura 9.5 Índice maestro para los dispositivos a los que se les ha hecho seguimiento.

indica que se suministró la ruta completa desde la raíz hasta el objeto. Por ejemplo, para hacer seguimiento a los objetos `ifInErrors` e `ifOutErrors` del dispositivo `10.0.2.200`, se usa el siguiente comando:

```
Target [10.0.2.200.1]:1.3.6.1.2.1.2.2.1.14.1&1.3.6.1.2.1.2.2.1.20.1\
:public@10.0.2.200
```

Este comando recupera tanto los conteos de errores de entrada como los de salida en la interfaz conocida como 1.

Nótese el uso del carácter **&** en el comando precedente. El MRTG requiere dos variables cuando presenta los datos; este carácter se usa como un separador. Como otra alternativa, se pueden usar los nombres de objetos SNMP abreviados, en lugar de la convención directa OID. Por tanto, el comando

```
Target[10.0.2.200.1]:ifInErrors.1&ifOutErrors.1:public@10.0.2.200
```

es funcionalmente equivalente a un comando previo **Target**. El MRTG conoce un número limitado de objetos SNMP; la lista de objetos se puede expandir modificando y añadiendo objetos al archivo `mibhelp.txt`. Este archivo contiene transformaciones entre algunos objetos MIB-II SNMP y rutas OID completamente calificadas. El siguiente es un ejemplo de este archivo:

```
Descriptor:     ifInErrors
Identifier:     1.3.6.1.2.1.2.2.1.14
ASN.1 Syntax:   Counter32

The number of inbound packets that contained
errors preventing them from being deliverable to a
higher-layer protocol.
```

Este ejemplo de archivo muestra el objeto `ifInErrors` que se usó en el comando **Target** precedente, y contiene un descriptor, un identificador y una sintaxis ASN.1 del objeto. El descriptor nombra al objeto que el MRTG usará, y está asociado con la cadena identificadora; puede ser cualquier nombre de cadena mientras sea única en el archivo `mibhelp.txt`. El identificador es el nombre de ruta completamente calificado del objeto, menos la instancia. La sintaxis ASN.1 se refiere al tipo de objeto y debería concordar con la definición del objeto que se encuentra en la MIB que define el objeto. Finalmente, la entrada contiene texto descriptivo que explica la definición del objeto. Aunque este texto no se requiere, es muy útil y su inclusión se recomienda cuando se agregan objetos adicionales a este archivo.

La tabla 9.6 presenta objetos SNMP que están predefinidos en el archivo `mibhelp.txt`. Consúltese el capítulo 6, "Vista general de MIB-II", para información adicional acerca de estos objetos MIB-II.

Tabla 9.6 Objetos predefinidos MRTG SNMP.

Objeto SNMP	Información del objeto
`ifOperStatus`	Estado operativo actual de la interfaz
`ifAdminStatus`	Estado administrativo actual de la interfaz
`ifInOctets`	Número total de octetos recibidos en la interfaz, incluidos caracteres de enmarcado

Tabla 9.6 Objetos predefinidos MRTG SNMP. (*continuación*)

Objeto SNMP	Información del objeto
ifInJcastPkts	Número de paquetes de subred de unidifusión entregados a un protocolo de capa superior
ifInNUcastPkts	Número de paquetes de no unidifusión (es decir, de difusión o de subred de multidifusión) entregados a un protocolo de capa superior
ifInDiscards	Número de paquetes entrantes que fueron escogidos para ser descartados
ifInErrors	Número de paquetes entrantes que contenían errores, evitando su entrega a un protocolo de capa superior
ifInUnknownProtos	Número de paquetes recibidos a través de la interfaz, que fueron descartados debido a un protocolo desconocido o no soportado
ifOutOctets	Número total de octetos transmitidos fuera de la interfaz, incluidos caracteres de enmarcado
ifOutUcastPkts	Número total de paquetes que los protocolos de nivel superior solicitan sean transmitidos a una dirección de subred de unidifusión, incluidos aquellos que fueron descartados o no se enviaron
IfOutNUcast Pkts	Número total de paquetes que los protocolos de nivel superior solicitan sean transmitidos a una dirección de no unidifusión (es decir, una subred de difusión o una subred de multidifusión)
ifOutDiscards	Número de paquetes salientes que fueron escogidos para ser descartados aunque no se haya detectado ningún error para evitar que sean transmitidos

Tabla 9.6 Objetos predefinidos MRTG SNMP. (*continuación*)

Objeto SNMP	Información del objeto
ifOutErrors	Número de paquetes salientes que no podrían ser transmitidos, debido a errores
ifOutQLen	Longitud de la cola de salida de paquetes (en paquetes)

El MRTG soporta operaciones matemáticas en los objetos SNMP o en cadenas OID. Estos operadores se podrían usar, por ejemplo, para agrupar el número total de errores de entrada y errores de salida para una colección de interfaces, en lugar de sólo una interfaz. Por tanto,

```
Target[monet]: ifInErrors.0:public@10.0.2.75+ifInErrors.1:public@10.0.2.75
&ifOutErrors.0:public@10.0.2.75+ifOutErrors.1:public@10.0.2.75
```

calcula la suma de los errores de interfaz para la primera y la segunda interfaz encontradas en el dispositivo 10.0.2.75.

Seguimiento del espacio del disco

El MRTG también soporta la colección de datos de programas o secuencias de comandos personalizadas. Por ejemplo, el MRTG puede presentar información del uso del espacio del disco. Usando la secuencia de comandos df2mrtg disponible en el directorio de software adicional, el MRTG hace un seguimiento a particiones individuales del disco y suministra información del uso y del tamaño del disco. El siguiente comando muestra esta tarea:

```
Target[monet]: '/usr/mrtg/bin/df2mrtg -u /dev/hdb2'
```

Colocando esta línea y algunas opciones adicionales en un archivo de configuración MRTG se produce el gráfico mostrado en la figura 9.6. La tabla 9.6 muestra que tener la capacidad de recolectar información de otras fuentes es una función muy poderosa del MRTG. Como se puede observar en la figura, el gráfico diario muestra el espacio del disco operando en aproximadamente 120 MB; luego, aproximadamente a las 6:00 PM (1800 horas), el espacio del disco se reduce casi a cero. Esto continuó por cuatro horas hasta que se liberó espacio adicional del disco después de las 10:00 PM de la misma noche, por encima de 180 MB. El gráfico mensual registra el mismo evento, pero los datos reales se

comprimen en una escala más pequeña. Por tanto, la ventana de cuatro horas sin espacio en el disco es sólo una corta caída en este gráfico. Estos gráficos facilitan precisar el consumo del disco por hora y por día y ayudarán bastante a determinar por qué el evento ocurrió en primer lugar.

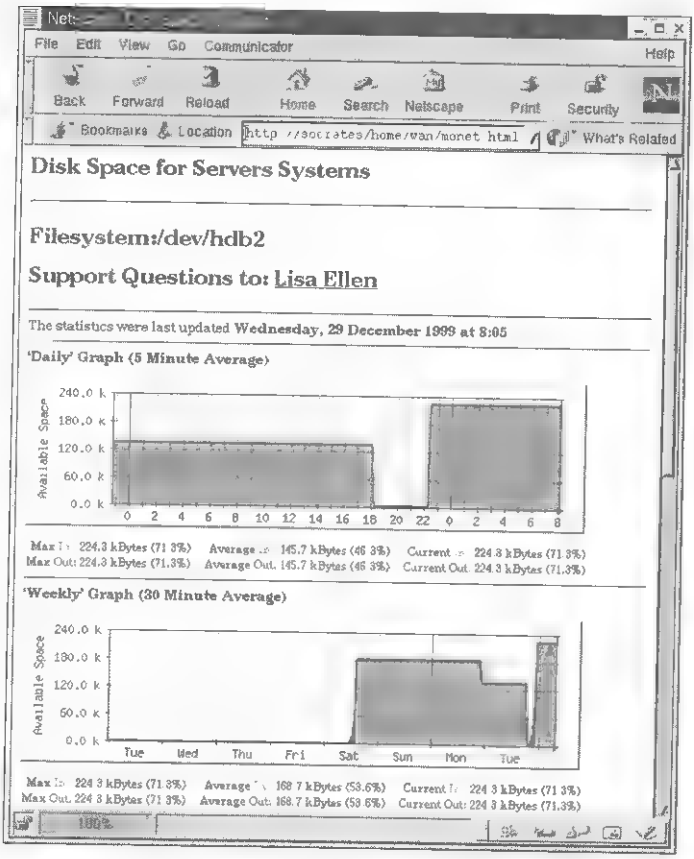


Figura 9.6 Reporte MRTG del espacio del disco.

Si se tiene interés en desarrollar programas personalizados para hacer un seguimiento a información diferente de la SNMP, los datos retornados del programa deben estar en un formato que el MRTG entienda. Por tanto, un programa debe retornar cuatro líneas de salida que incluyan el estado presente de la primera variable, el estado presente de la segunda variable, una cadena indicando el tiempo arriba del sistema objetivo y una cadena indicando el nombre del objetivo.

Las primeras dos variables son comparables, por ejemplo, con los bytes entrantes y bytes salientes por segundo. El tiempo arriba del sistema es indicativo de cuándo se reinició el sistema por última vez (o cuándo se inició por última vez el componente de administración de la red, el agente). Para asegurarse de que la salida de la secuencia de comandos concuerda con un objetivo, se retorna el nombre del objetivo.

En el caso de la secuencia de comandos `df2mrtg`, la primera variable representa el espacio del disco en kilobytes que se usa actualmente (cuando se usa la opción `u`), y la segunda variable es una repetición de la primera. Debido a que sólo se tiene interés en hacer un seguimiento al espacio del disco, no son necesarias la información de tiempo arriba y la cadena que representa el objetivo. Además, dependiendo del tipo de datos retornados con la secuencia de comandos, se puede necesitar usar la palabra clave **Options** ya sea con `gauge` o `absolute`, como sea apropiado. Esta secuencia de comandos también reporta otra información respecto a las particiones del disco del sistema local. La tabla 9.7 presenta las opciones de línea de comandos.

Tabla 9.7 Opciones de línea de comandos de la secuencia de comandos **df2mrtg**.

Opción	Descripción
`-f`	Reporta el espacio libre del disco en la partición
`-s`	Reporta el tamaño total de la partición del disco
`-p`	Reporta el espacio libre del disco como un porcentaje
`-u`	Reporta el espacio de disco usado

Si se quiere que el MRTG grafique el espacio libre del disco en una partición, se usa la opción `f`. La opción `s` reportará el tamaño total de la partición.

Debido a que el MRTG se usa para hacer seguimiento a interfaces de red, deberían cambiarse algunos de los valores predeterminados e información de configuración, cuando se use

para hacer seguimiento al espacio del disco o a otra información del sistema. Enseguida se presenta la configuración MRTG que esta más cercana a la que es apropiada cuando se usa la secuencia de comandos `df2mrtg`.

```
Title[monet]: Disk Space Monitor
PageTop[monet]: <H1>Disk Space Usage for System:Monet
Options[monet]: gauge,growright
MaxBytes[monet]: 314461
AbsMax[monet]: 400000
WithPeak[monet]: ymwd
YLegend[monet]: Available Space
ShortLegend[monet]: Bytes
Legend1[monet]: Current Disk Utilization.
```

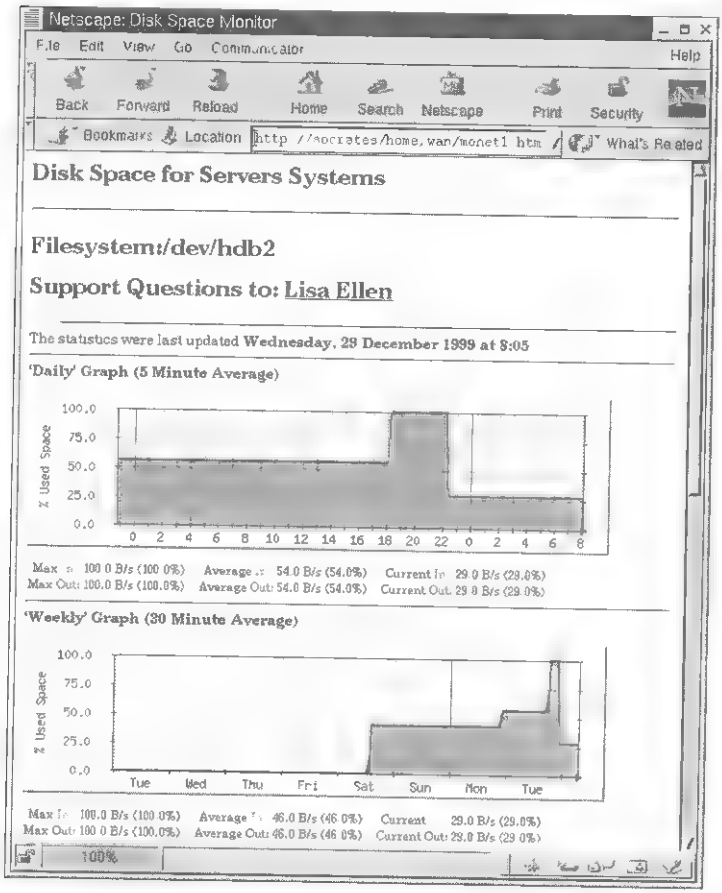


Figura 9.7 Reporte MRTG de uso del disco, en porcentajes.

Debido a la flexibilidad del MRTG y la secuencia de comandos `df2mrtg`, en lugar de reportar espacio disponible del disco, se podrían reportar porcentajes del espacio disponible del disco. Ésta es tal vez una mejor medida porque, presentando el porcentaje de utilización, se puede obtener un estado rápido de la condición del espacio disponible del disco, sin operaciones matemáticas. La figura 9.7 muestra la misma partición, pero utiliza la opción `-p` de la secuencia de comandos para reportar en porcentaje de disco. Como se puede observar, este gráfico muestra la misma información que la figura 9.6, pero presentar esta información como un porcentaje es más significativo y fácil de leer.

Componentes adicionales para el MRTG

En el paquete MRTG se han incluido varios módulos con el fin de suministrar soporte para personalización adicional del MRTG y hacer seguimiento de servicios y dispositivos adicionales. La tabla 9.8 suministra una lista de estos módulos, que se encuentran en el directorio `/usr/lib/mrtg/contrib`.

Tabla 9.8 Módulos adicionales para el MRTG.

Nombre del módulo	Descripción
`ascendget`	Suministra información del módem del dispositivo de acceso remoto de Ascend
`atmmaker`	Construye archivos de configuración MRTG para dispositivos ATM
`cpumon`	Hace un seguimiento al uso de la CPU de un sistema UNIX
`distrib`	Construye gráficos de distribución de tráfico de la red, dados los archivos de registro MRTG
`get-active`	Produce un reporte de las interfaces más activas a las que el MRTG les hace seguimiento
`get-multiserial`	Hace un seguimiento a los módems de acceso telefónico (*dial-in*) en un sistema local UNIX

Tabla 9.8 Módulos adicionales para el MRTG. (*continuación*)

Nombre del módulo	Descripción
`GetSNMPLinesUP`	Consulta dispositivos SNMP para objetos específicos y regresa valores al MRTG
`ircstats`	Recolecta información en clientes activos IRC
`mrtg-archiver`	Archiva los GIF del reporte MRTG, dada una lista dispositivos objetivo
`mrtg-blast`	Incorpora la salida `tcp-blast` en el MRTG
`mrtg-dgnip`	Habilita el MRTG para hacer seguimiento a las interfaces dinámicas de red
`mrtg ipget`	Habilita el MRTG para hacer seguimiento a interfaces expresadas con direcciones IP, en lugar de instancias
`mrtg-mail`	Hace un seguimiento de las estadísticas de correo
`mrtgidx`	Genera un índice MRTG de dispositivos a los que se les hace seguimiento
`mrtgindex.cgi`	Genera un índice MRTG de dispositivos a los que se les hace seguimiento (una secuencia de comandos mejorada)
`ping-probe`	Presenta información del rendimiento de un viaje de ida y vuelta, usando **ping**
`portmasters`	Presenta información de un módem de dispositivos Livingston Portmaster
`rdlog2`	Produce presentaciones de tráfico de dispositivos a los que el MRTG les hace seguimiento

Tabla 9.8 Módulos adicionales para el MRTG. (*continuación*)

Nombre del módulo	Descripción
`rumb-stat`	Obtiene estadísticas de bajo nivel de servidores de red como Apache
`TCH`	Hace seguimiento a la utilización del módem de los *racks* de módems USR Total Control
`xlsummary`	Extrae información resumida de los archivos de registro MRTG y produce gráficos de Excel

La instalación y operación de estas herramientas se encuentra en el directorio que lleva el mismo nombre (o similar) que el de la secuencia de comandos.

Resolución de problemas MRTG

Debido a que el archivo de configuración MRTG consiste en texto sencillo, es relativamente simple introducir errores de sintaxis u otros errores en el archivo durante la personalización. Además, pueden existir otras condiciones que produzcan errores cuando se ejecutan otros programas MRTG. Enseguida se presentan algunos de los problemas más comunes de tiempo de ejecución y de configuración, y las soluciones propuestas.

Mensaje de error/problema:

```
Rateup WARNING: .//rateup could not read the primary log file for
   10.0.2.50.1
Rateup WARNING: .//rateup The backup log file for 10.0.2.50.1 was
   invalid as well
Rateup WARNING: .//rateup Can't remove 10.0.2.50.1.old updating log
   file.
```

Solución

No hay por qué preocuparse acerca de estos errores. Éstos ocurren de vez en cuando pero no plantean ningún problema serio y se pueden ignorar. Desafortunadamente, no hay forma de inhabilitar la generación de estos mensajes.

Mensaje de error/problema:

```
No answer from community@device. You may be using the wrong community
```

Solución

El MRTG tiene problemas al sondear un dispositivo con la cadena de comunidad SNMP especificada en el archivo de configuración. Se debe revisar para asegurarse de que se está usando la cadena de comunidad correcta para el(los) dispositivo(s) que tienen problemas. Además, asegúrese de que el(los) dispositivo(s) se puede(n) sondear desde la red usando otra herramienta SNMP como `snmpget` o `snmpwalk`.

Mensaje de error/problema:

```
./mrtg rmon
ERROR: I guess another mrtg is running. A lockfile (rmon_1) aged
264 seconds is hanging around. If you are sure that no other mrtg
is running you can remove the lockfile.
```

Solución

El MRTG detectó un archivo de bloqueo, el cual indica que otra copia del MRTG está corriendo actualmente o fue corrida en el pasado y falló al tratar de remover el archivo de bloqueo. En este caso se usa el comando **ps** para asegurarse de que ningún proceso adicional MRTG está corriendo. Si ningún proceso adicional está corriendo, remueva el archivo de bloqueo.

Mensaje de error/problema:

```
ERROR: I can't find a "target[10.0.2.50.1]" definition
```

Solución

El MRTG reporta que el comando de palabra clave **target** hace falta en el archivo de configuración. Adicione la entrada correcta en el archivo de configuración.

Mensaje de error/problema:

```
ERROR: "WorkDir" not specified
ABORT: Please fix the error(s) in your config file
```

Solución

El comando **WorkDir** no se incluyó en el archivo de configuración que se especificó con el comando **mrtg**. Debido a que esta palabra clave no es opcional, el MRTG aborta sin esta definición.

Mensaje de error/problema:

```
Bareword found where operator expected at (eval 13) line 1, near "1public"
(Missing operator before public?)
Bareword "public" not allowed while "strict subs" in use at (eval 13) line
1. Array found where operator expected at (eval 13) line 1, at end of line
* Problem with '1public@10.0.2.50': syntax error at (eval 13) line 1, near
"1public"
```

Solución

Se detectó un error de sintaxis en el archivo de configuración MRTG. Se debe revisar la línea indicada en el mensaje de error en busca de dos puntos que faltan, u otros errores de sintaxis.

```
Target[10.0.2.50.1]:1public@10.0.2.50
```

Mensaje de error/problema:

```
SNMP Error:
no response received
SNMPv1_Session (remote host: "10.0.2.200" [10.0.2.200].161
                  community:  "public"
                 request ID:  472779887
               PDU bufsize:   8000 bytes
                    timeout:  2s
                    retries:  5
                    backoff:  1)
SNMPGET: Failed to reach target: "1:public@10.0.2.200". I
tried multiple times!
```

Solución

Este error indica que el dispositivo no respondió a las solicitudes SNMP. En este caso se debe hacer una revisión para asegurarse de que el dispositivo está activo en la red y que el agente SNMP está respondiendo.

Herramienta ntop

El programa ntop suministra una forma de determinar los sistemas que están activos en la red y los tipos de tráfico enviados y recibidos por estos sistemas. El tráfico de la red es ordenado por protocolo, y alguno de los protocolos de red más comunes son

reconocidos, incluidos TCP/IP y protocolos asociados. Esta herramienta también soporta tipos de protocolos diferentes del IP, como Decnet, IPX, AppleTalk y otros. Recuérdese el uso de `etnereal` para capturar e inspeccionar paquetes de red individuales, analizado en el capítulo 5, "Utilidades y herramientas adicionales del sistema". Esta herramienta difiere significativamente de las utilidades `tcpdump` o `etnereal`, en que se centra en suministrar datos estadísticos acerca de los paquetes de red, no de sus contenidos. Además, **ntop** no necesita el uso de un servidor Web; éste soporta el protocolo HTTP internamente.

¿Por qué usar **ntop**? En primer lugar, éste suministra una forma rápida y fácil de conseguir una instantánea precisa de la actividad de la red sin usar una sonda de red o un dispositivo husmeador (*sniffer*). En muchos casos, se necesita una sonda dedicada para localizar problemas de red. En ciertas situaciones, sin embargo, hay que actuar rápidamente y **ntop** se puede usar cuando no es posible obtener una sonda rápidamente, porque está instalada y haciendo seguimiento a otra red. Segundo, puede no ser posible conectar una sonda, dadas ciertas configuraciones de red, por ejemplo, entre un par de sistemas UNIX interconectados a través de una WAN. En esta situación, usar **ntop** cuando se instala una sonda puede ser muy difícil, si no imposible. Tercero, mientras el costo sea una consideración importante, comprar sondas será mucho más costoso que instalar una herramienta de software de dominio público.

En el lado negativo, ¡nada es gratuito! Correr **ntop** a través de varios sistemas plantea algunos problemas. Primero, **ntop** coloca la interfaz en modo promiscuo, lo cual significa que se está capturando cada paquete de red, y como resultado, se coloca en el sistema una carga de procesamiento significativa. Para solucionar este problema, se debería tener cuidado cuando se ejecuta **ntop**. Configurar algunos mecanismos automatizados puede remediar cualquier problema potencial, asegurando que **ntop** se corre solamente cuando es necesario y, tal vez, no durante los tiempos críticos o el uso pico del sistema.

En segundo lugar, usar **ntop** puede complicar además las políticas y los procedimientos de seguridad, porque el acceso no autorizado a esta herramienta podría generar exposición adicional a vulnerabilidades y agujeros de seguridad del sistema existentes. No debe tomarse una idea equivocada acerca de la herramienta **ntop**; ésta no introduce ningún problema nuevo de seguridad o vulnerabilidad en los sistemas existentes. Más bien, suministra

información adicional en cualquier sistema instalado en la red. Afortunadamente, **ntop** tiene una facilidad rudimentaria para proteger la información de red capturada. Ésta no es la seguridad más rigurosa disponible, pero suministra una cantidad razonable de control. Esto se analizará más adelante.

En tercer lugar, **ntop** no soporta todos los protocolos de trabajo en red conocidos por el ser humano. De hecho, para una herramienta de dominio público, su cubrimiento es impresionante. Éste suministra un conjunto útil de protocolos que se usan en la mayoría de las redes hoy en día. Éstos incluyen IP, IPX, NetBIOS y Decnet. También suministra los protocolos de mayor nivel TCP/IP, como FTP, HTTP, DNS, Telnet y otros.

Una de las características más poderosas de **ntop** es que suministra dos modos de vista: modo Web y modo terminal básico ASCII. Ésta es una característica muy exclusiva, y **ntop** es una de las herramientas de dominio público que implementa exitosamente múltiples y robustas capacidades de interfaz. Sin embargo, las características más interesantes se encuentran usando la interfaz Web.

En el modo Web, está disponible una serie de páginas HTML que suministra varios reportes estadísticos de tráfico de red e información adicional relacionada con la red. En este modo se necesita un explorador estándar Web para ver estos reportes y para soportar múltiples y simultáneas conexiones de explorador. Ésta es una característica muy práctica cuando se usa **ntop** en un entorno de red distribuido donde el acceso se puede originar desde más de una ubicación por diferentes usuarios al mismo tiempo. Cuando un explorador está conectado a **ntop**, se presenta la página principal que se muestra en la figura 9.8.

Uso del modo Web

Para iniciar **ntop** en el modo Web, se usa la opción `-w` seguida de un número de puerto. Éste es el puerto que los exploradores Web clientes usarán para conectarse con el programa **ntop**. Por tanto, para iniciar **ntop** en el puerto 3000, se emite el siguiente comando:

```
# ntop -w 3000
```

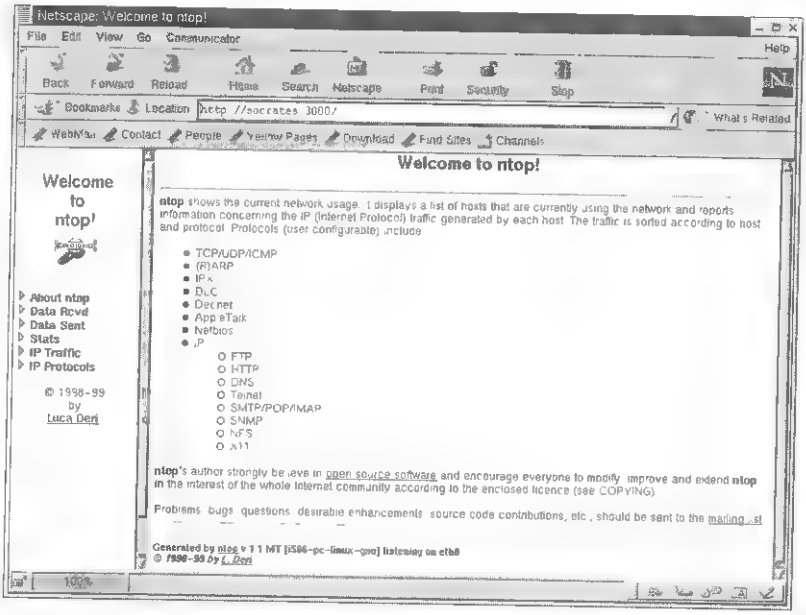


Figura 9.8 Página Web principal `ntop`.

Después de que la herramienta ha arrancado, presenta la siguiente salida:

```
ntop v.1.1 MT [i586-pc-linux-gnu] listening on eth0.
Copyright 1998-99 by Luca Deri <deri@unipi.it>
Warning: unable to read file '.ntop'. No security will be used!
Waiting for HTTP connections on port 3000...
```

ntop muestra cuál interfaz está usando, cuando más de una está instalada en el sistema, y confirma el puerto especificado que se usa para conexiones entrantes por exploradores clientes. Además, advierte que no pudo leer el archivo de seguridad y, como resultado, no se usará ninguna seguridad. En este caso, cualquier explorador cliente puede conectarse a **ntop** y ver los reportes. La seguridad de los reportes se cubrirá después.

Con el modo Web, la página principal se divide en dos marcos separados, como se muestra en la figura 9.8. El marco del lado izquierdo suministra una lista de categorías de reporte que el usuario puede seleccionar, mientras que el de la derecha presenta un reporte de tráfico de red. Las categorías incluyen Acerca de ntop (*About ntop*), Datos recibidos (*Data Rcvd*), Datos enviados (*Data Sent*), Estadísticas (*Stats*), Tráfico IP (*IP Traffic*) y Protocolos

IP (*IP Protocols*). Cada una de éstas contiene uno o más elementos (o reportes) seleccionables que suministran información detallada relacionada con la categoría especificada. El usuario debe pulsar en una categoría usando el botón izquierdo del ratón para exponer los elementos contenidos en esa categoría. Cada una de las categorías importantes (y reportes subsecuentes) se describen enseguida.

Las pantallas usadas a lo largo de esta sección se han tomado directamente de un explorador Netscape mientras se usa la aplicación **ntop**. Se consideró que la integridad real de la salida (la cual es un tema importante a lo largo de este libro) debe mantenerse a toda costa. Sin embargo, cada uno de los reportes se guardó primero en un nuevo marco HTML. Esto generó nuevas ventanas, que se crean para presentar los reportes. Con Netscape, pulsar el botón derecho genera un menú pequeño, y uno de los elementos del menú suministra la capacidad de guardar cada reporte usando un nuevo marco. La razón más importante para hacer esto es que cada reporte puede estar contenido en una única ventana. Desafortunadamente, ésta es una espada de dos filos. Cuando se crean nuevos marcos, se remueve el panel izquierdo. Como resultado, la vista del árbol de cada reporte no se presenta junto con cada reporte asociado. Además, algunos de los reportes son significativamente más grandes que el tamaño máximo de la ventana del explorador y deben presentarse en dos marcos diferentes.

Categoría de datos recibidos (*Data Rcvd*)

Este grupo de reportes muestra las estadísticas actuales de tráfico de red para datos entrantes a todos los dispositivos de red locales del sistema que corre la herramienta **ntop**. Los reportes incluyen:

- Reporte de todos los protocolos
- Reporte IP
- Reporte de rendimiento
- Reporte de flujos de red

Reporte de todos los protocolos (All Protocols Report)

Este reporte muestra el tráfico de red recibido por los *hosts* de red listados, ordenados por el nodo de red o el tipo de protocolo de red. Este reporte presenta una tabla grande, como se muestra

en la figura 9.9. La figura muestra los *hosts* activos, la cantidad de datos recibidos en kilobytes, el porcentaje de tráfico de red recibido por cada sistema y la cantidad de tráfico de red perdido por cada tipo de protocolo. El protocolo listado representa el más activo de la red. Para cambiar el tipo de orden predeterminado, simplemente se pulsa sobre uno de los nombres de título. Esto forzará a la tabla a redibujarse con base en la nueva selección.

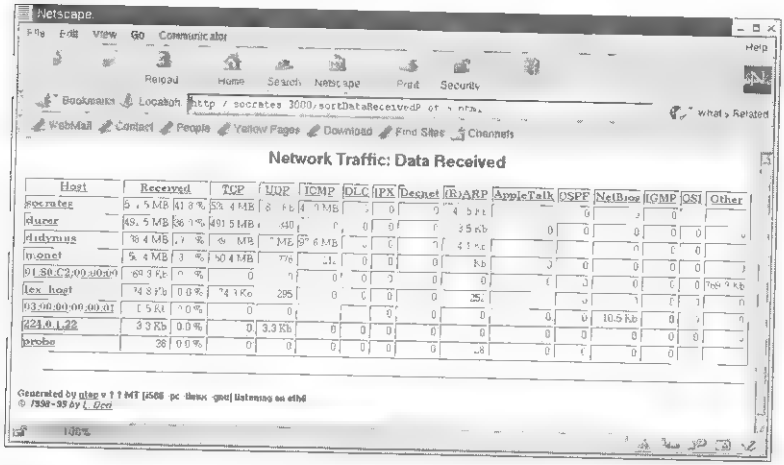


Figura 9.9 Reporte de todos los protocolos.

En este ejemplo, se han identificado varios *hosts*, incluidos `socrates`, `durer`, `rembrandt` y `224.0.1.22`. Éstos son los sistemas activos que reciben tráfico actualmente en la red. Sin embargo, en el caso de `224.0.1.22`, éste representa un grupo de multidifusión, y otros sistemas en la red envían tráfico periódicamente a esta dirección. Si la multidifusión está habilitada en la red, se pueden presentar otras direcciones de multidifusión en otros reportes **ntop**. Además, **ntop** suministra un reporte de multidifusión específico que se analiza después.

Este reporte muestra la cantidad total de tráfico recibido en la segunda columna, expresada en bytes, kilobytes (`KB`), megabytes (`MB`) o gigabytes (`GB`). Además, la columna *received* (recibido) muestra, como un porcentaje, la cantidad de tráfico que recibió cada *host*. En este reporte, de los sistemas listados, `socrates` fue el receptor con mayor cantidad de tráfico, el cual representa `41.8%` del total del tráfico obtenido en la red. Las columnas restantes muestran la cantidad de tráfico recibido, basada en cada uno de los protocolos de red soportados por **ntop**. Las columnas

tercera y cuarta muestran la cantidad total de datos que eran paquetes TCP o UDP. En este caso, TCP aportó una gran cantidad de tráfico.

Reporte IP (IP Report)

Este reporte usa el mismo formato de los reportes de Protocolos IP, pero además define los protocolos TCP/IP que se registraron para incluir funciones de aplicación de mayor nivel, como la transferencia de archivos, acceso remoto y acceso Web. La figura 9.10 muestra este reporte.

Las columnas incluyen los datos totales recibidos por protocolos como FTP, Telnet, HTTP y otros. Aunque éste reporte contiene sólo un pequeño número de host, desde el punto de vista del rendimiento de la red éste proporciona información muy útil. Se puede suponer que el protocolo FTP se usa más que cualquier otro protocolo listado. Esto se determina comparando el tráfico total FTP con todos los otros protocolos.

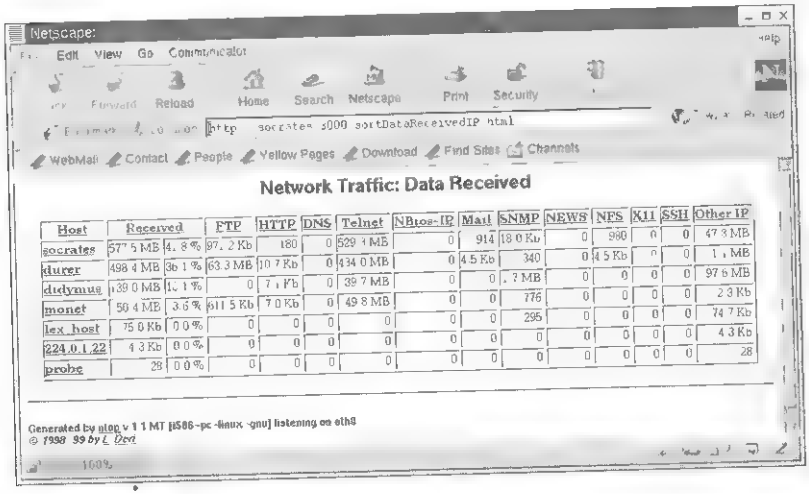


Figura 9.10 Reporte IP.

Reporte de rendimiento (Throughput Report)

La figura 9.11 muestra este reporte, el cual representa una excelente forma de determinar el rendimiento real de los dispositivos individuales en la red. El rendimiento se define como el número de bytes de tráfico de red enviados durante un periodo especificado. Este reporte suministra tres columnas: Actual Thpt, Average Thpt (promedio) y Peak Thpt (pico). La columna Actual

Tnpt muestra el rendimiento de cada sistema durante los últimos tres segundos. Realmente, hay una pequeña demora entre el momento en que los datos del gráfico se recolectaron y presentaron y el momento en que el explorador se actualizó por última vez. Por tanto, los resultados están desactualizados durante unos pocos minutos y no están completamente en tiempo real. La columna Ave rage Tnpt muestra el rendimiento de cada sistema promediado durante el periodo en que **ntop** estaba corriendo. La columna Peak Thpt representa cualquier pico en los patrones de tráfico. De modo predeterminado, la tabla se ordena en la columna Ac tual Thpt.

Como se puede observar en la tabla, el anfitrión socrates tiene un gran pico de 289.7Kpbs, pero el promedio era mucho más pequeño. Cuando se mide o determina el rendimiento de la red, es mejor tomar los números pico de rendimiento con un poco de suspicacia y tener en cuenta otros factores. Es decir, los picos solamente muestran cuánto se usó de la red en el punto más alto durante el tiempo en que se le hizo seguimiento a la red. Lo mejor es considerar también los promedios, porque representan la verdadera naturaleza de uso de la red durante un periodo más largo. Los picos son temporales y fluctúan considerablemente, mientras que los promedios reflejan un comportamiento regular.

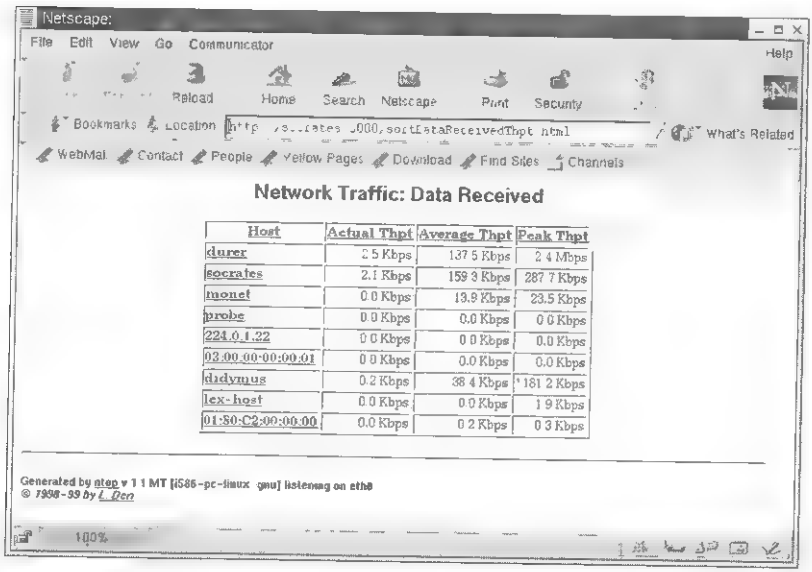


Figura 9.11 Reporte de rendimiento.

Reporte de flujos de red (Network Flows Report)

Una característica muy interesante de **ntop** es su habilidad para definir flujos personalizados de red (es decir, filtros) y presentar los resultados. Básicamente, un flujo es una expresión de filtro (como aquellos analizados en el capítulo 4, "Herramientas y utilidades centrales del sistema", con `tcpdump`) que se especifica en la línea de comandos mediante la opción `-F`. Por ejemplo, el comando

```
# ntop -F "servermonitor='host durer',session='host socrates\
and monet'" -w 3000
```

define dos flujos diferentes. El primero filtra el tráfico si ni la dirección de la fuente ni de destino es igual a `durer`. El otro captura el tráfico entre `socrates` y `monet`. Los nombres de los flujos son cadenas simples y pueden contener cadenas de caracteres normales ASCII. Al ejecutar este comando se presentará el reporte que se muestra en la figura 9.12. La figura muestra ambos flujos e incluye conteos totales de paquete y la cantidad total de bytes transmitidos o recibidos. Se puede crear cualquier número de flujos, y el **ntop** soporta muchas de las palabras clave de expresión de filtro y la sintaxis que soportan `tcpdump` y `etnereal`. Éstas se presentaron en el capítulo 4, "Herramientas y utilidades centrales del sistema", y en el capítulo 5, "Utilidades y herramientas adicionales del sistema", respectivamente. Es claro que los flujos suministran una forma poderosa para proporcionar rendimiento al seguimiento. La sección titulada "Configuración de flujos" analiza la creación de flujos.

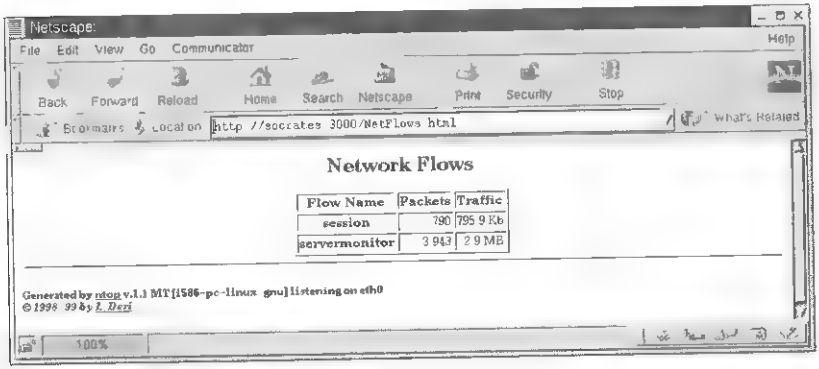


Figura 9.12 Reporte de flujos de red.

Categoría de envío de datos (*Data-sent*)

Estos reportes muestran las estadísticas reales de tráfico de red para datos salientes de todos los dispositivos de red locales del sistema que corre la herramienta **ntop**. Los reportes incluyen:

- Reporte de todos los protocolos
- Reporte IP
- Reporte de rendimiento

Reporte de todos los protocolos (All Protocols Report)

Este reporte, que se muestra en la figura 9.13, enumera los *hosts* que transmitieron tráfico en la red. El formato de este reporte es el mismo del reporte recibido, excepto que contiene una columna `Sent` en lugar de una columna `Received`. Este reporte es útil para localizar *hosts* que generan o reciben cantidades significativas de paquetes. En algunos casos, esto podría representar una situación normal que refleja patrones verdaderos de tráfico; también podría indicar algún tipo de problema. Esto podría suministrar evidencia de un dispositivo mal configurado o de alguna otra actividad no deseada. Por ejemplo, es interesante notar que el *host* `durer` transmitió un cierto número de paquetes `IPX` en la red. A pesar de que sólo una cantidad relativamente pequeña de datos se envió usando este protocolo, esto podría ser una razón para sospechar que ocurre un problema, porque la red a la que se le hace seguimiento debería contener sólo tráfico basado en el IP. Realmente, es común tener sistemas PC que transmitan protocolos como `IPX` y `NetBios`, aunque el servicio que utiliza estos protocolos no se esté usando. Además, el número de paquetes ICMP parece bastante grande. Esto podría indicar que una gran cantidad de solicitudes ping (u otros mensajes ICMP) están inundando la red.

Reporte IP (IP Report)

Este reporte se parece al reporte IP de la categoría de *datos recibidos* (`Data Recd`) analizada en la sección previa de reporte. Es interesante notar que en esta tabla, la actividad base TCP/IP muestra que `Telnet` representa la porción más grande de 494.6 megabytes de datos transmitidos, con `FTP` en un distante segundo lugar, con sólo 40.3 megabytes. La figura 9.14 muestra un ejemplo de reporte. Estos tipos de reportes pueden ser útiles cuando se necesita establecer cuáles aplicaciones se usan más.

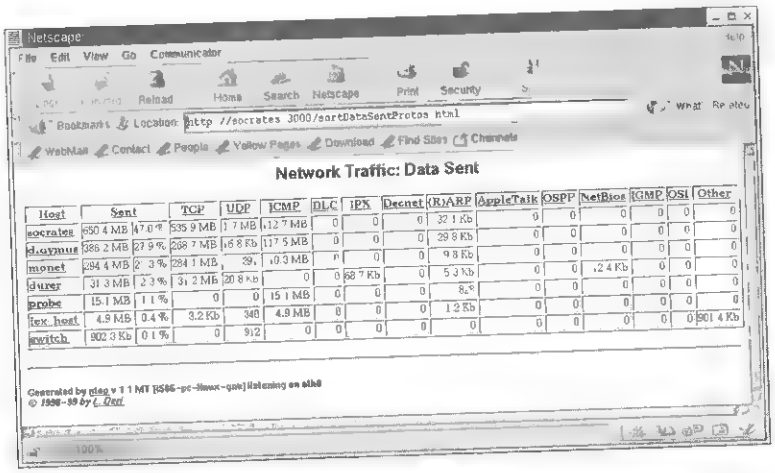


Figura 9.13 Reporte de todos los protocolos.

Para muchas compañías, el acceso HTTP o Web puede ser el generador de tráfico más significativo. Como se puede observar, es muy fácil determinar cuáles aplicaciones se usan más. Si se está usando un protocolo no estándar en la red, pero **ntop** no lo soporta directamente, aún es posible adicionar un filtro específico de protocolo y hacer un seguimiento a la actividad del protocolo con un flujo. Consúltese la sección de reporte previa acerca de cómo configurar flujos.

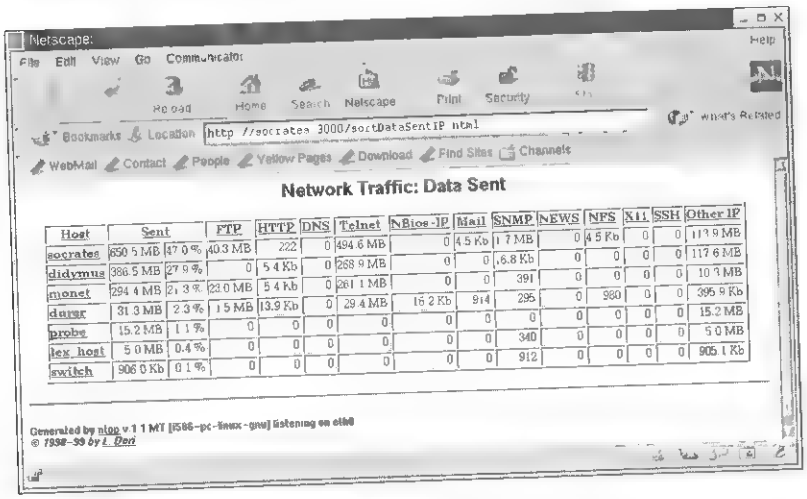


Figura 9.14 Reporte IP.

Reporte de rendimiento (Throughput Report)

El reporte de rendimiento de datos enviados tiene el mismo formato que el reporte de rendimiento de datos recibidos. Sin embargo, este reporte puede indicar mucho, cuando se trata de conocer realmente qué es lo que está sucediendo en una red. La transmisión de tráfico puede identificar el pulso de la red, mucho más que la información de datos recibidos. Esto es cierto porque, cuando se miden los datos recibidos, la información puede venir de otras fuentes que no sea la red local. Ésta también es una métrica de rendimiento importante, pero tener la capacidad de identificar a *quienes más usan la Web* o a los *usuarios avanzados* en la red es difícil.

Considérese la tabla de la figura 9.15, donde el *host* `socrates` fue el responsable de una cantidad significativa del tráfico en la red. Nótese que éste tiene un gran pico durante el periodo de seguimiento. Es importante notar que el `Actual Thpt` actual descendió considerablemente. Esto es bueno. Si los niveles pico se han sostenido durante un mayor periodo, debería hacerse un examen más a fondo de las posibles causas. Además, no es una gran sorpresa que `socrates` pudiera transmitir el pico mostrado. Con muchos servidores y estaciones de trabajo poderosos, es un asunto relativamente simple exceder las limitaciones de la red, en particular con sistemas de alta calidad que contienen múltiples CPU. En este ejemplo, se supone que la red era una Ethernet normal de 10 Mbps. En este caso, `socrates` consumió cerca de `25%` de la red. Sin embargo, si se está hablando de la Ethernet Rápida (*Fast Ethernet*, 100 Mbps), sólo se usó `2.5%` de la red.

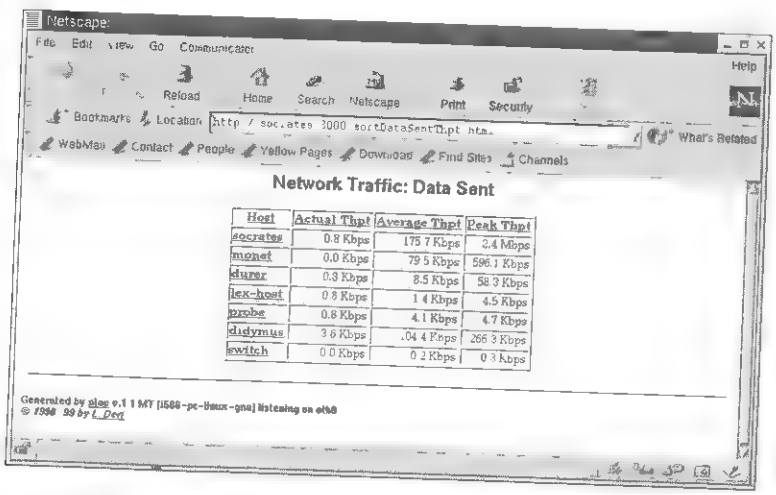


Figura 9.15 Reporte de rendimiento.

Categoría de estadísticas (*stats*)

Estos reportes muestran las estadísticas generales de tráfico y otra información importante respecto al *host* en la red. Esta categoría contiene los siguientes reportes:

- Reporte de estadísticas de multidifusión
- Reporte de estadísticas de tráfico
- Reporte de *hosts*
- Reporte de rendimiento

Reporte de estadísticas de multidifusión (Multicast)

Este reporte muestra toda la actividad de paquete de multidifusión en la red. Con multidifusión, se definen direcciones IP especiales (como `224.0.0.1`), y sistemas que suministran servicios de multidifusión, como Internet TV, transmitirán datos utilizando estas direcciones especiales. Además, la información acerca de cuáles dispositivos pertenecen a cuáles grupos de multidifusión también se propaga usando una dirección de multidifusión estándar. El reporte de multidifusión revela cuáles *hosts* están transmitiendo información de multidifusión. Ciertos sistemas, como las estaciones de trabajo y los servidores Sun, continuarán sondeando direcciones específicas de multidifusión para determinar si está definido algún grupo de multidifusión. Como resultado, estos sistemas generarán tráfico, a pesar de que ninguna aplicación de multidifusión está activa en esos sistemas.

En general, para soportar la multidifusión, la infraestructura de red debe configurarse de tal modo que pueda soportar grupos de multidifusión que estén propagando y reenviando tráfico de multidifusión. Si la red no está configurada para soportar específicamente la multidifusión y los reportes indican una cantidad de tráfico significativa, entonces la investigación está garantizada. Si la multidifusión se ha configurado, será importante determinar qué tráfico de multidifusión está dirigido sólo a aquellas redes y sistemas que desean esta información. En la práctica, esto se obtiene de manera automática, porque la activación del enrutado de multidifusión maneja muchos de los detalles necesarios para asegurar que los paquetes de multidifusión se envíen correctamente. Sin embargo, errores de configuración u otros problemas pueden llevar a que la red se vea inundada por información de multidifusión que no se desea. En este caso, se usa el reporte de multidifusión para localizarlos.

La figura 9.16 muestra una vista instantánea de otra red conmutada bastante grande que contiene muchos sistemas. La tabla incluye muchos dispositivos que han transmitido pequeñas cantidades de tráfico de multidifusión. Sin embargo, algunos sistemas han enviado mucho más. Un ejemplo es el dispositivo `cisco-gw2`. Éste es el enrutador local predeterminado en el que el sistema **ntop** estaba corriendo, y el que ha transmitido una cantidad de tráfico significativa, como se muestra en la figura. El único problema es que no se ha configurado ninguna estación de trabajo en la subred para soportar la multidifusión. Como resultado, la estación de trabajo en la red local ha solicitado tráfico de multidifusión o el enrutador se ha configurado mal. Puede surgir la siguiente pregunta: "¿Por qué se ven tantos dispositivos en el reporte?" La respuesta es que cuando se envía un paquete usando una de las direcciones de multidifusión, esta información se transmite a través de la red a todas las subredes que se han configurado (o configurando de manera impropia) para enviar esta información. Como consecuencia, el reporte muestra todos los *hosts* que han transmitido datos, sin importar su localización en la red. En el caso de los *hosts* UNIX que transmiten información de multidifusión que no se desea, el tráfico se puede bloquear (no ser enviado por los enrutadores locales) o inhabilitar en la fuente, apagando uno o más de los servicios de la red.

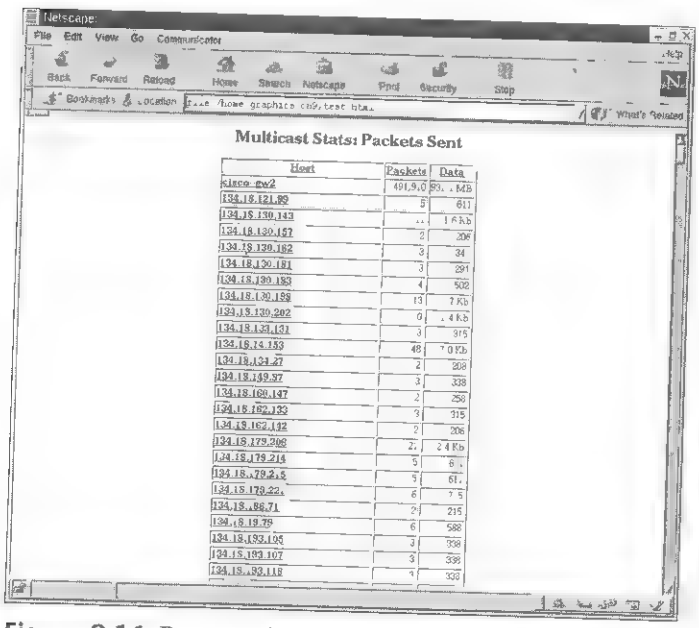


Figura 9.16 Reporte de estadísticas de multidifusión.

Reporte de estadísticas de tráfico (Traffic)

El reporte de estadísticas de tráfico global contiene información interesante relacionada con el rendimiento general de la red desde un punto de vista de nivel superior. Este reporte contiene dos tablas: la primera (que se muestra en la figura 9.17) ofrese información global de tráfico, mientras que la otra (que se muestra en la figura 9.18) ofrece información sobre la distribución de la utilización de protocolo.

En la primera tabla, las primeras tres filas contienen información genérica. Con tantos sistemas UNIX hoy en día, que tienen conexiones con redes múltiples, es muy común y fácil hacer seguimiento a la red equivocada. El campo `Nw Interface Type` (tipo de interfaz) revela cuál interfaz está sondeando el programa **ntop**. Como se indicó previamente, es importante asegurarse de que se le está haciendo seguimiento a la red correcta. La tabla muestra que se usó la primera interfaz, `eth0`. El campo `Local Domain Name` (nombre de dominio local) muestra el nombre del dominio para el sistema que corre **ntop**. Nótese que éste no es un dominio válido, pero es un ejemplo.

El campo `Sampling Time` (tiempo de muestreo) es muy importante, ya que muestra la cantidad de tiempo que ha estado activo el programa **ntop**. Es importante recordar que, con el fin de obtener la información más precisa acerca del rendimiento de la red, a ésta debe hacérsele seguimiento tanto como sea posible. Esto no quiere decir que a una red deba hacérsele seguimiento durante un periodo fijo, pero sí suficiente para establecer cualquier patrón común de uso u otros eventos interesantes o especiales. Por ejemplo, si se programa una tarea de red en particular una vez a la semana y se quiere capturar el impacto de esta tarea en la red, entonces debería hacérsele seguimiento a la red antes y después de ocurrido el evento. También es importante hacerle un seguimiento a la red durante el tiempo suficiente, para establecer una línea base o referencia. La línea base es una medida que representa la conducta normal de la red y se compara con la actividad anormal. En muchas redes, hacer seguimiento por una semana o dos debería ser suficientemente adecuado para determinar la línea base. La tabla muestra que **ntop** ha estado corriendo por cerca de 8.5 horas. Este periodo se puede usar para echar un vistazo al rendimiento de la red, pero se recomienda un periodo más largo de seguimiento.

La siguiente sección de la tabla muestra algunos detalles interesantes acerca del tráfico recolectado. La estadística `Total`

representa el número total de marcos capturados en la interfaz a los que **ntop** ha estado haciendo seguimiento desde que fue iniciado. El campo `Dropped by the kernel` indica los marcos abandonados debido a limitaciones de procesamiento desde un punto de vista UNIX. Algunas veces, el sistema (es decir, el kernel UNIX) descartará paquetes debido a limitaciones de la CPU u otras limitaciones temporales. El campo `Dropped by` **ntop** muestra el número de paquetes descartados por **ntop** debido a problemas con el paquete. Cualquier campo puede indicar que el sistema no podría procesar los paquetes de modo suficientemente rápido, o que se definieron uno o más filtros (o flujos), lo cual originó que **ntop** no usara el paquete.

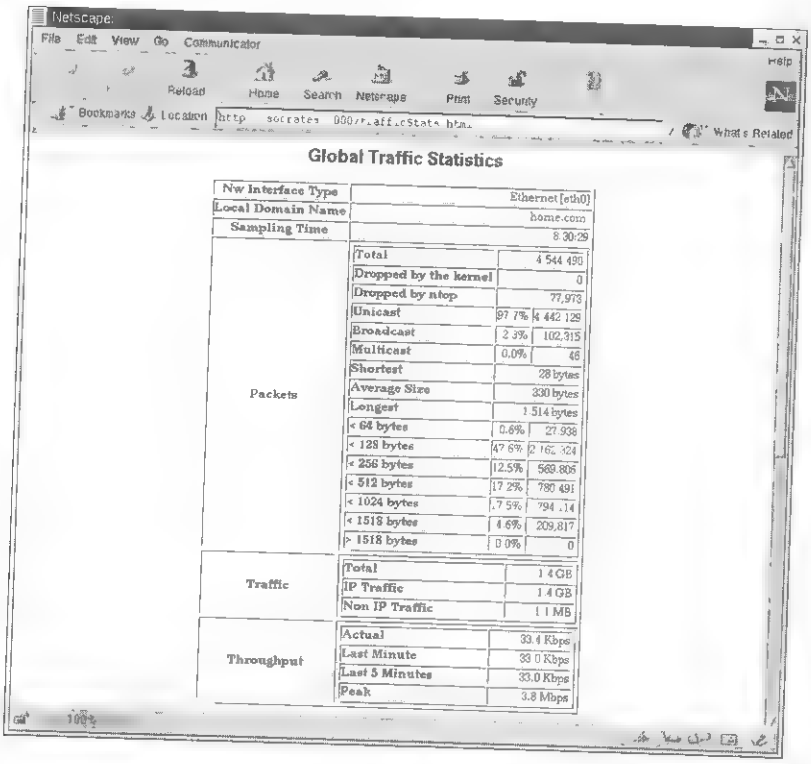


Figura 9.17 Primer reporte de estadísticas de tráfico.

Las siguientes tres filas presentan los tres tipos de dirección de paquete comunes registrados y sus porcentajes asociados. Ellos incluyen `Unicast` (unidifusión), `Broadcast` (difusión) y `Multicast` (multidifusión). El tipo de paquete revelará, en algún grado, cómo se está usando la red. Por ejemplo, si la mayoría del

tráfico se basa en la multidifusión, entonces podría concluirse que las aplicaciones de multimedia son la fuente primaria del tráfico de red. Sin embargo, es importante examinar muy cerca el porcentaje. Si un porcentaje grande de tráfico se basa en la difusión o en la multidifusión pero no hay aplicaciones de red que lo generen, entonces debe resolverse un misterio. Esto es fácil, porque otros reportes **ntop** muestran esta información. En general, las redes deberían contener mayormente tráfico de unidifusión. El tráfico de unidifusión está representado por una máquina que envía información a otra. De nuevo, la red y la instalación de aplicaciones determinarán el perfil del tráfico, en última instancia.

Las filas restantes de la tabla perfilan el tamaño de los paquetes registrados, y no deberían pasarse por alto. Hace algunos años, una gran corporación estaba instalando una nueva aplicación que involucraba el uso de una interfaz gráfica de usuario y de una base de datos en la trastienda (*back-end*). Como la aplicación se estaba instalando en sitios remotos alrededor del mundo, el personal de desarrollo y soporte notó graves problemas de rendimiento cuando se usaba la nueva aplicación. El problema: la aplicación estaba usando paquetes muy pequeños entre la interfaz de aplicación y la base de datos. El problema se agravó aún más por el hecho de que la red introdujo una cierta cantidad de latencia para cada paquete. Como resultado, el tiempo de respuesta de la aplicación se podía medir en minutos, no en segundos. El problema se debió a que la aplicación estaba usando sólo una pequeña fracción del ancho de banda de red disponible. La solución, en este caso, demandó el cambio de la interfaz de la aplicación, para solicitar más información por paquete. Esto, finalmente, solucionó el problema, pero el comportamiento se descubrió de manera tardía en el ciclo de desarrollo, lo cual causó demoras en la instalación de la aplicación y generó gastos adicionales de desarrollo.

La tabla de la figura 9.17 muestra que 47% de los paquetes contenían menos de 128 bytes cada uno. Además, el promedio de ejecución fue sólo de 330 bytes. Esto es desafortunado, porque el tamaño de paquete máximo soportado por la red es mayor de 1,500 bytes. En muchas situaciones, el tamaño del paquete utilizado y la naturaleza de muchas aplicaciones y servicios UNIX estándar no pueden alterarse con facilidad. Por tanto, las mejoras con los servicios estándar UNIX no serán significativas. De otro lado, los grupos de desarrollo de aplicaciones personalizadas deberían poner más atención a esta información

y, de esta forma, detectar a tiempo problemas relacionados con la red en el ciclo de vida del desarrollo del software.

Una observación adicional: en muchos entornos WAN, el tamaño máximo de paquete es generalmente más pequeño que el que está disponible en la mayoría de tecnologías de red local (LAN). Esto también debe tenerse en cuenta.

La siguiente sección, `Traffic` (tráfico), muestra totales generales de la actividad de la red. El campo `Total Traffic` (tráfico total) indica la cantidad total de datos que se leyeron desde la interfaz. Ésta corresponde a la suma tanto del tráfico de red basado en el IP como del no basado en IP. El campo `IP Traffic` (tráfico IP) muestra la cantidad total de tráfico IP, el cual incluirá, por ejemplo, todos los protocolos de nivel superior, como ICMP y TCP. La estadística de `Total non-IP Traffic` (tráfico diferente del IP) incluye protocolos como IPX, AppleTalk y otros. Por tanto, si se tiene una red mezclada de protocolos IP y AppleTalk, este campo no será cero y contendrá la porción de tráfico AppleTalk. Desafortunadamente, parece que la aplicación es inexacta respecto a esos totales, porque los campos `IP Traffic` y `non-IP Traffic` deberían igualar el valor presentado en el campo `Total`. Las unidades de tráfico son diferentes; para IP se usan GB, mientras que para no IP se usan MB. Luego, con los datos mostrados en la gráfica, los valores están bien aproximados.

La sección `Throughput` (rendimiento) mide el rendimiento observado de la interfaz (recepción de paquetes); las secciones de reporte anteriores describen estas estadísticas.

Este reporte contiene dos tablas adicionales, las cuales se muestra en la figura 9.18. La primera tabla, titulada `Global Protocol Distribution` (Distribución global de protocolo), muestra la cantidad total de datos (kilobytes) enviados y recibidos por cada familia principal de protocolos. Por ejemplo, la cantidad total de datos para el protocolo `IP` fue 1.4Gb, mientras que el `IPX` tenía sólo 69.8 Kb. El reporte suministra una subdivisión adicional de protocolo en la categoría `IP` que incluye `TCP`, `UDP` e `ICMP`, con valores asociados de protocolo. Estos valores también se presentan a la derecha de la tabla, en un formato de gráfico de barras que refleja los porcentajes de cada protocolo con respecto al `IP` total. Nótese que el `TCP` representa 81.0% del tráfico `IP`, mientras que el `ICMP` representa 18.9%. El `UDP` esta vacío porque el valor (desde el punto de vista del porcentaje) es menor que uno y no se podría presentar, dada la escala del gráfico. Cuando se visualiza en el explorador, las barras del gráfico

son de color azul claro. En las figuras asociadas, aparecen como una sombra de color gris claro.

La segunda tabla muestra el tráfico total IP ordenado por diferentes protocolos TCP/IP asociados. La primera columna muestra el tipo de protocolo y la segunda columna muestra la cantidad total de datos para ese protocolo. El lado derecho incluye la representación visual de los datos como un porcentaje. La vista de los porcentajes muestra la cantidad de tráfico para cada protocolo. Como se puede observar, el protocolo más activo fue `Telnet`, seguido de otros protocolos `TCP/IP` no específicos. Entre los protocolos y/o servicios que pueden estar en esta categoría están `remote`, `login`, `finger`, `DNS` y otros como éstos.

Reporte de estadísticas de hosts (*Hosts*)

Este reporte suministra información adicional relacionada con los *hosts* que se hallaron en la red. Incluye las siguientes columnas: `Host Name` (nombre del *host*), `IP Address` (dirección IP), `MAC Address` (dirección MAC), `Sent Bandwidth` (ancho de banda enviado) y `Nw Board Vendor` (vendedor de tarjeta de red). La figura 9.19 contiene un ejemplo de este reporte. Si la dirección IP de un *host* no se puede transformar en un nombre de *host*, se usa la dirección IP (`IP address`) en la columna de `Host Name`. Ésta es la razón por la que algunas de las entradas contienen el nombre de *host* y la dirección IP. El campo de dirección IP se explica a sí mismo, y la columna de la dirección `MAC` (control de acceso al medio) contiene la dirección de hardware de la interfaz del *host* que envió/recibió tráfico de red. La columna `Sent Bandwidth` suministra una vista del ancho de banda enviado para cada *host* en la red. Éste se calcula mediante el uso de la suma de todo el tráfico enviado (no recibido) a cada *host* y graficando los valores asociados.

La columna final contiene el vendedor del hardware de la interfaz, como se deduce de los primeros tres bytes del campo `MAC Address`. Nótese que no todas las entradas contienen un vendedor de interfaz, y la `224.0.1.22` no contiene ninguna dirección MAC en absoluto. La razón es simple: los dos *hosts* que tienen direcciones MAC pero no vendedores asociados son direcciones de bajo nivel relacionadas con un servidor, usadas para anunciar servicios específicos. La `224.0.1.22` es una dirección de multidifusión que no corresponde a ninguna dirección específica de vendedor, pero sí a una o más direcciones MAC específicas y estándar. Sin embargo, el software **ntop** no detectó estas direcciones.

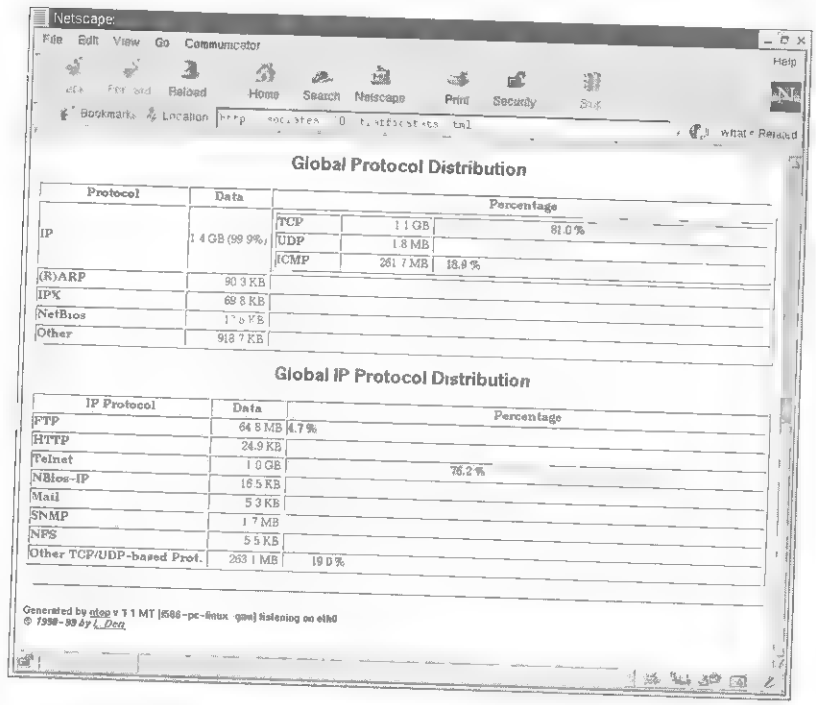


Figura 9.18 Segundo reporte de estadísticas de tráfico.

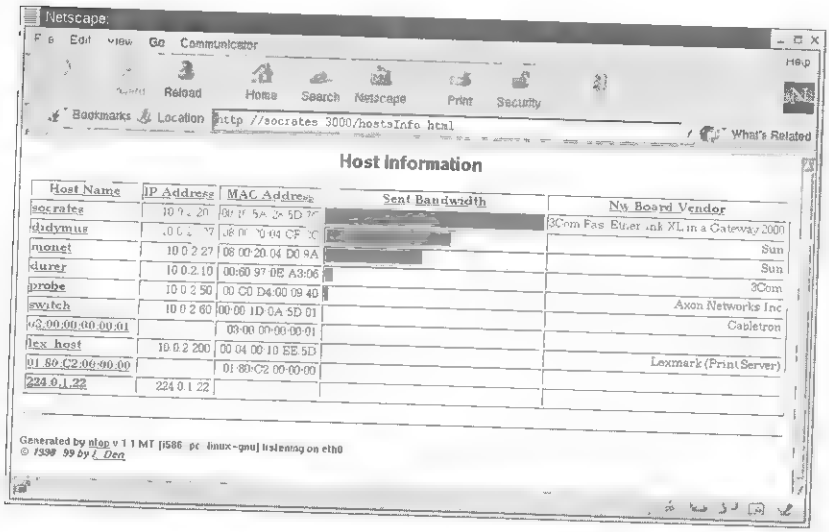


Figura 9.19 Reporte de *host*.

Nótese que las entradas en la columna `Host Name` están subrayadas. Esto se debe a que representan un enlace a otro reporte **ntop**. Los otros títulos están subrayados también, pero con un propósito diferente. Éstos se usan para seleccionar el orden de cada columna. El reporte del nombre del *host* (`Info about`) suministra información adicional sobre el *host* e incluye información de tráfico más detallada, historia IP y una lista de sistemas iguales (*peer*) contactados. La figura 9.20 muestra este reporte para el *host* `socrates`. Como se puede observar, alguna de la información presentada en el reporte también se puede encontrar en el reporte `Host Info`.

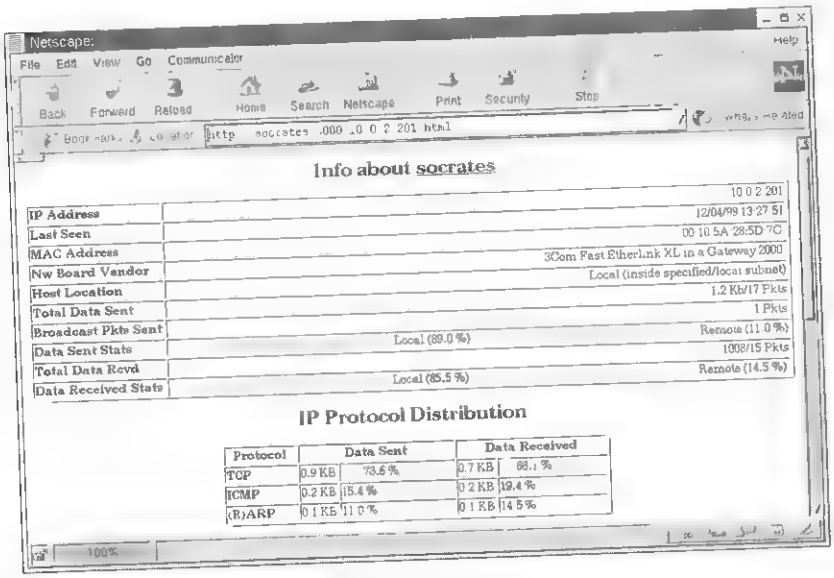


Figura 9.20 Reporte de *host* individual.

Los primeros tres elementos ya se analizaron. La `Host Location` (localización del *host*) indica que este *host* está contenido en la subred a la que **ntop** está haciendo seguimiento. En otras palabras, la interfaz desde donde **ntop** está haciéndole seguimiento está en la misma red que el *host*. ¿Por qué esto es importante? Porque ayuda a saber si este *host* es local o remoto, desde la perspectiva de **ntop**, por tanto la información adicional que reporta **ntop** es interpretada correctamente. Por ejemplo, el campo `Data Sent Stats` (estadísticas de datos enviados) da una representación gráfica de los porcentajes tanto del tráfico local como del remoto. Esta información se puede usar para determinar

patrones de tráfico a través de una internet. En el caso del reporte mostrado aquí, 100% del tráfico es local. Si esta situación, por ejemplo, ha sido reversada, y 100% del tráfico fuera remoto, podría ser necesario mover socrates a otra red remota o limitar el acceso a este sistema desde una localización remota. Total Data Sent (total de datos enviados) representa tanto el número de bytes como el número de paquetes transmitidos desde socrates.

La estadística Total Data Sent es una combinación de tráfico de difusión y multidifusión expresada tanto en bytes como en número de paquetes enviados. Ésta es una medida verdadera de lo que este *host* transmitió en la red, con respecto al destino final de la información. La estadística Broadcast Pkts Sent (paquetes de difusión enviados), como su nombre sugiere, indica el número de paquetes que usaron la dirección de difusión (255) para transmitir paquetes. Esto incluye los paquetes de difusión tanto IP como de enlace de datos. Estos tipos de paquetes se usan para enviar información a todos los dispositivos locales en la red. El campo Data Received Stats (estadísticas de datos recibidos) indica que todo el tráfico recibido fue local del sistema.

Las tablas restantes incluyen IP Service/Port Usage (uso del puerto/servicio IP), IP Session History (historia de la sesión IP) y Active TCP Sessions (sesiones activas TCP), como se muestra en la figura 9.21. La sección IP Session (Sesión IP) del reporte incluye actividad IP con la cual estuvo involucrado este *host*. En este caso, se registraron tanto el tráfico TCP como el UDP. El reporte también suministra detalles adicionales acerca del protocolo de nivel superior involucrado en la actividad de la red, es decir, superior que el TCP o el UDP. Como se puede observar, tanto los servicios ftp como telnet se listan, y se suministra información adicional respecto a estos servicios. En la primera tabla, la actividad ftp que se muestra incluye el número de sesiones (veces) para las cuales el servicio estaba activo, los bytes enviados/recibidos, e información del sello de tiempo.

La tabla Active TCP Session (sesión activa TCP) incluye gran parte de la información que se puede obtener del comando **netstat**. Sin embargo, suministra un formato más fácil para revisar la información que la salida de este comando. En general, proporciona una forma de ver el estado de todas las conexiones basadas en el TCP tanto desde la perspectiva del cliente como desde la del servidor. La columna Client Port muestra el sistema remoto y el puerto TCP utilizado, el cual representa el extremo remoto de la conexión. El Server Port, que es el extremo

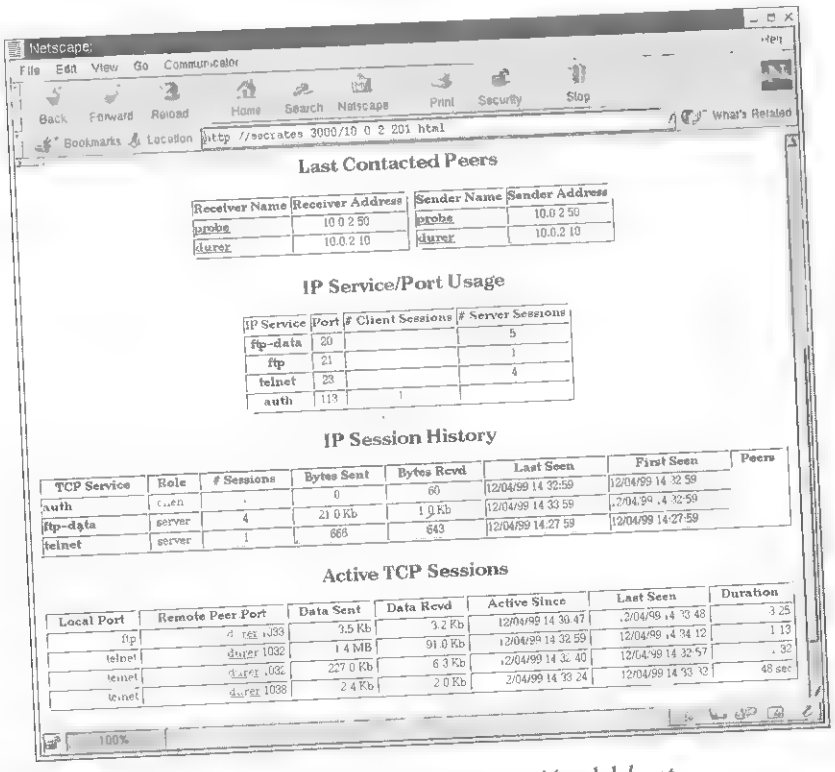


Figura 9.21 Reporte adicional de información del *host*.

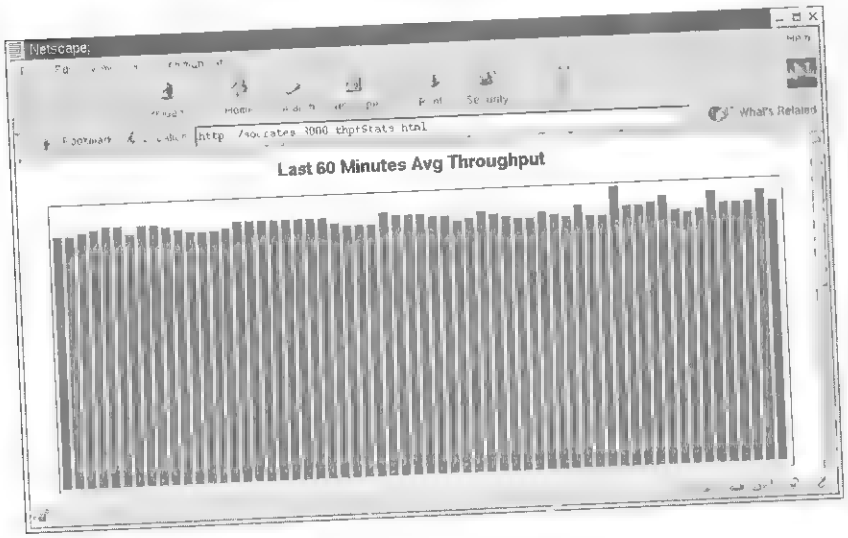


Figura 9.22 Reporte de rendimiento.

local de la conexión, también muestra el sistema y el puerto TCP. En el capítulo 2, "Protocolo simple de administración de red", se analizó cómo una conexión TCP consta de cuatro elementos: la dirección remota IP, el puerto remoto, la dirección IP local y el puerto remoto.

Las columnas `Data Sent` (datos enviados) y `Data Rcvd` (datos recibidos) muestran la cantidad de datos enviados/recibidos por la conexión. El campo `Active Since` (Activa desde) muestra cuándo (fecha y hora) fue iniciada la conexión por primera vez. El campo `Duration` (duración) también muestra por cuánto tiempo estuvo activa la conexión. Es interesante notar que cuando una conexión ya no está activa desde un punto de vista TCP (en el modo `TIME_WAIT`, por ejemplo), la duración se incrementa hasta que la conexión está completamente terminada y ya no aparece en el reporte.

Nótese también que, cuando es posible, **ntop** presenta el nombre de la aplicación de servicio, en lugar de presentar sólo el puerto TCP. Por ejemplo, el reporte muestra que varias sesiones `telnet` estuvieron activas entre los sistemas `durer` y `rembrandt`. El reporte presenta el puerto como `telnet`, no como el número de puerto normal asociado con la aplicación `Telnet` (es decir, `23`). **ntop** puede hacer esta transformación porque examina cada uno de los puertos TCP que encuentra e investiga el archivo `/etc/services` para una correspondencia apropiada.

Reporte de rendimiento (Throughput)

Este reporte contiene gráficos que muestran el rendimiento promedio de todos los dispositivos en la red a los que se le está haciendo seguimiento. Es un resumen del tráfico tanto para la actividad de los paquetes recibidos como de los enviados, comparado con el ancho de banda de la red. El gráfico que se muestra en la figura 9.22 es el primer gráfico que se presenta en este reporte. El gráfico muestra la última hora de seguimiento, y cada barra representa un minuto. La escala del gráfico no es proporcional; el fondo es cero, pero el tope no es 100%. El reporte también incluye otro gráfico que representa el promedio de las últimas 24 horas de ejecución. La barra más alta representa el pico para ese periodo.

Categoría de tráfico IP (*IP Traffic*)

Los siguientes cinco reportes presentan el tráfico de red en más detalle, con un énfasis particular en la relación entre los dispositivos en la red. Los reportes incluyen remoto a local (`R->L`), local

a remoto (L->R), local a local (L<->L), Matriz de tráfico IP (Matrix) y uso local por el proceso UNIX (Local Usage), desde la perspectiva del sistema que corre al **ntop**. Los primeros tres reportes comparten el mismo reporte general diseño/formato y los dos restantes son distintivos.

Reportes remoto y local

La figura 9.23 muestra el reporte de Remote to local IP Traffic (tráfico IP de local a remoto). Como muchas de las otras tablas, este reporte lista nombres de *hosts* específicos e información de dirección IP. El reporte también suministra enlaces para obtener información adicional del *host*, como se describió previamente. Éstos también incluyen dos tablas separadas, una que muestra el tráfico individual para cada *host*, y otra que muestra los totales para todos los *hosts* combinados. En la primera tabla, las estadísticas Data Received (datos recibidos) y Data Sent (datos enviados) contienen dos columnas. La primera columna muestra la cantidad de datos recibidos o enviados por cada sistema, expresados en bytes. La segunda columna muestra la cantidad de tráfico como un porcentaje para todos los *hosts* listados en la tabla. La segunda tabla contiene la columna Total Traffic (tráfico total), la cual es un resumen de las columnas Data Received y Data Sent de la primera tabla. Las siguientes dos columnas, Data Received y Data Sent, son el resumen de las columnas con el mismo nombre en la primera tabla. La columna Bandwidth (ancho de banda) lista la utilización efectiva del ancho de banda como resultado del tráfico de red al que se le hace seguimiento.

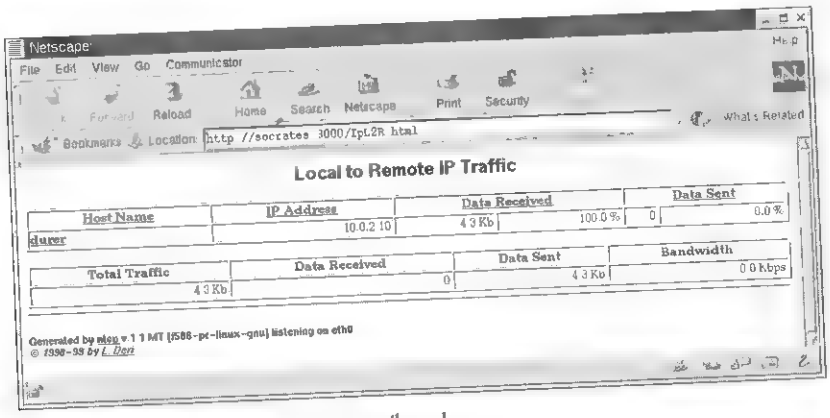


Figura 9.23 Reporte IP remoto/local.

Los dos reportes restantes suministran información similar, pero desde el punto de vista del tráfico enviado/recibido de los sistemas locales a las redes y sistemas remotos y a sistemas locales que se comunican con otros sistemas locales. Queda para el lector exportar y revisar estos reportes adicionales.

Reporte de matriz de tráfico (Matrix)

Este reporte revela la cantidad total de tráfico que va y viene desde y hacia cada sistema. La figura 9.24 presenta un ejemplo del reporte de matriz de tráfico. El esquema de color está basado en si excede el tráfico y en cuánto excede los umbrales de rendimiento listados en la tabla 9.9. Cada uno de los sistemas es comparado con el valor más grande que consta tanto de los valores enviados como recibidos por ese sistema. Dependiendo del tráfico, a cada sistema se le asigna un color apropiado en sus respectivas celdas. En la figura, la celda que interseca los *hosts* `socrates` y `durer` es roja. Debido a que este libro fue producido en blanco y negro, los colores aparecen como sombras de color gris oscuro. Esta celda también contiene la cantidad total de datos pasados entre ambos sistemas. Para mantener la simetría, las celdas se repiten porque los *hosts* se intersectan cada dos veces.

Tabla 9.9 Umbrales de rendimiento del reporte de matriz de tráfico.

Color	Descripción
`rosado`	Entre 0 y 15% del valor máximo
`verde`	Entre 15 y 75% del valor máximo
`rojo`	Por encima del 75% del valor máximo

Este reporte es útil por las siguientes razones. Primero, es muy fácil determinar cuáles sistemas se están comunicando entre sí. Esto es importante cuando el tráfico de red excede las limitaciones de la red local y los dispositivos se deben particionar para reducir la sobrecarga de la red e incrementar el rendimiento. Conocer los patrones de tráfico es el primer paso en la administración efectiva del rendimiento. Segundo, suministra totales entre sistemas para determinar los mayores generadores de tráfico. Por ejemplo, `socrates` y `durer` produjeron la mayoría del tráfico entre todos los otros sistemas. Desde la perspectiva de la administración, es

importante determinar por qué estos dos sistemas generaron esta cantidad de tráfico. Esto podría ser algo normal o podría ser un inconveniente; todo depende de cuáles servicios y/o aplicaciones se ofrecen en estos dos sistemas. Usando este reporte y otros, es relativamente fácil determinar qué tipo de servicios se están usando y en qué medida.

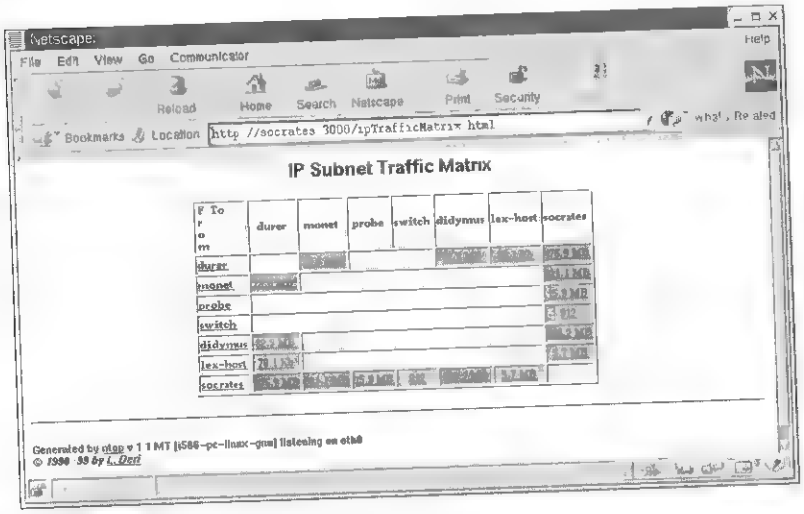


Figura 9.24 Reporte de matriz de tráfico.

Una nota adicional acerca de este reporte: cada una de las celdas intersecadas entre los mismos sistemas está en blanco y no contiene ningún valor. Éste es el resultado del seguimiento que **ntop** hace a una interfaz de red específica que normalmente nunca ve tráfico de sí misma. Cuando un proceso de red UNIX quiere comunicarse con un sistema, puede escoger usar la interfaz de ciclo de retorno. Como resultado, cuando **ntop** está haciendo seguimiento de interfaces reales, nunca ve este tráfico. Si se está interesado en hacerle seguimiento a la actividad en la red interna (es decir, ciclo de retorno), se usa la opción `-i` para cambiar la interfaz predeterminada cuando se invoca **ntop**. Puede causar sorpresa el hecho de que sólo una pequeña cantidad de *hosts* (la dirección de ciclo de retorno y la dirección primaria de interfaz) serán listados en los reportes cuando se captura tráfico de la red interna.

Reporte de uso local (Local Usage)

Este reporte contiene dos secciones: la primera muestra el uso de la red que hace cada proceso del sistema y la otra muestra el uso de la red por puerto.

Este reporte es muy útil porque suministra la capacidad de sondear cuáles aplicaciones y servicios del sistema están generando y recibiendo tráfico de la red en un sistema dado. Por supuesto, **ntop** debe estar corriendo en el sistema local para que esta información sea recolectada, pero bien vale la pena instalar esta herramienta a través de los sistemas requeridos para obtener esta información. El reporte también incluye información adicional útil, como la identificación del proceso, el usuario dueño del proceso y conteos de tráfico recibido y enviado. La figura 9.25 muestra un reporte de ejemplo de la utilización de la red solamente por cada proceso. Como se puede observar, la tabla de reporte fue truncada en la figura, debido a su longitud, la cual puede abarcar varias instantáneas de la pantalla si se presenta completa. La tabla es así de larga porque un típico servidor UNIX generalmente contiene una cantidad considerable de procesos que hacen uso de la red.

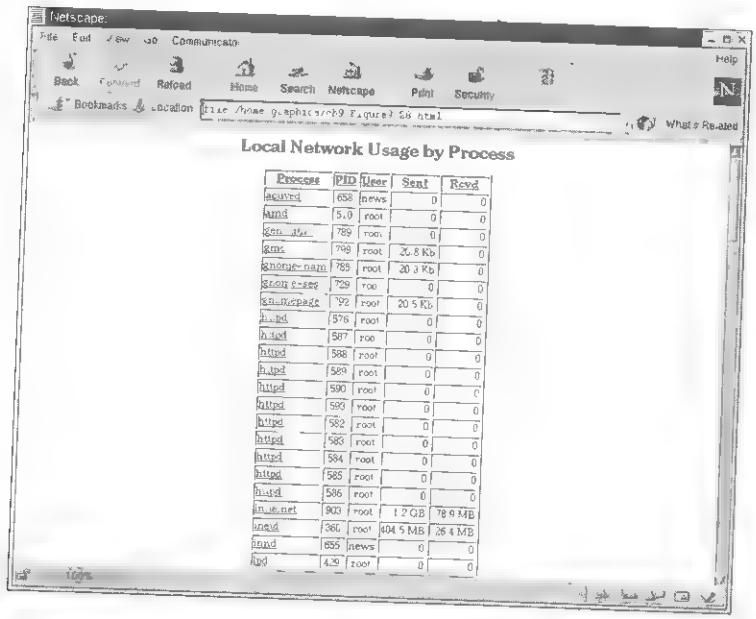


Figura 9.25 Reporte del uso local (parte uno).

La otra parte del reporte muestra una transformación entre el puerto de red y el proceso UNIX que usa ese puerto. Esto es interesante porque suministra una forma de establecer exactamente cuáles procesos usan determinados puertos de red. Parte de esta información se puede obtener del archivo `/etc/services`, pero no todas las aplicaciones o servicios de red se listan en este archivo, ni cada entrada revela exactamente cuál proceso puede usar un puerto en particular. La figura 9.26 presenta un ejemplo del reporte de la transformación.

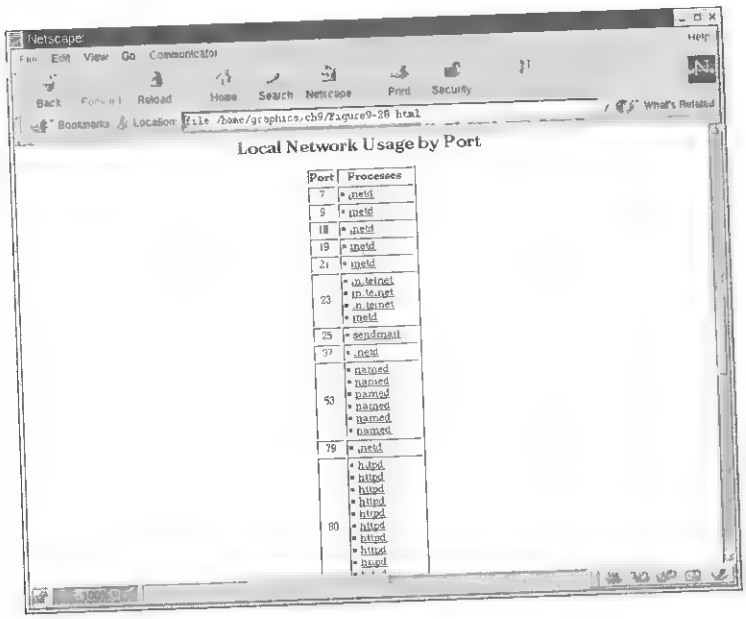


Figura 9.26 Reporte del uso local (parte dos).

Categoría de protocolo IP (IP Protocol)

En esta categoría se incluyen dos reportes diferentes. El primero, llamado `Distribution` (distribución), muestra hasta tres tablas separadas indicando el flujo del tráfico. El segundo, `Usage` (uso), muestra el uso de protocolo de red desde un punto de vista de alto nivel.

Reporte de distribución (Distribution)

Uno de los más útiles de todos los reportes **ntop** es el reporte de distribución del protocolo IP, el cual muestra la repartición del uso de la red por los protocolos de mayor nivel. Es similar al reporte

de `estadísticas globales de tráfico` que se mostró en la figura 9.17, pero contiene una repartición detallada por protocolos de la capa de transporte. Éste revela la cantidad de datos (enviados/recibidos) por tipo de protocolo, como `snmp` o `ftp`, según muestra la figura 9.27.

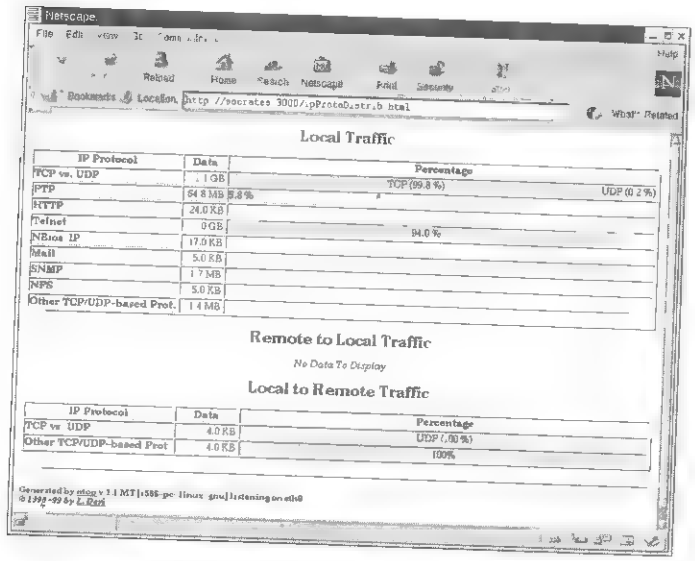


Figura 9.27 Reporte de distribución de protocolo.

Ésta es una observación importante porque este reporte muestra claramente cuáles aplicaciones son las más usadas frente a aquellas que se usan menos. El reporte contiene tres tablas que incluyen protocolo de distribución para tráfico total (`total traffic`), tráfico de remoto a local (`remote to local traffic`) y tráfico de local a remoto (`local to remote traffic`). Cada sección o tabla contiene tres columnas: `Protocolo IP`, `Datos` y `Porcentaje`. La columna `Protocolo IP` incluye los protocolos de nivel superior que **ntop** entiende y soporta. Aquellos protocolos que el **ntop** no soporta se listan bajo la categoría `Otner ICP/UDP-based Prot`. Cada protocolo tiene columnas asociadas de `Datos` y `Porcentaje` que presentan la cantidad de datos, expresados en bytes y como un porcentaje de bytes totales, en comparación con cada uno de los otros protocolos.

El uso de los gráficos de barras contenidos en este reporte es relativamente sencillo para identificar la utilización de la red por protocolo o aplicación de red. Por ejemplo, la segunda tabla

muestra que aparte de todo el tráfico IP presentado, el uso del `UDP` fue, sin duda, el más significativo al compararlo con el tráfico `TCP`.

Este reporte muestra protocolos de alto nivel o servicios de aplicación, y tanto el cliente como el servidor que participaron en el intercambio de tráfico. La figura 9.28 presenta un reporte de ejemplo que muestra `ftp`, `telnet`, `snmp`, `finger` y `smtp`. También muestra la lista tanto de los clientes como el servidor que estaban usando estos protocolo y servicios.

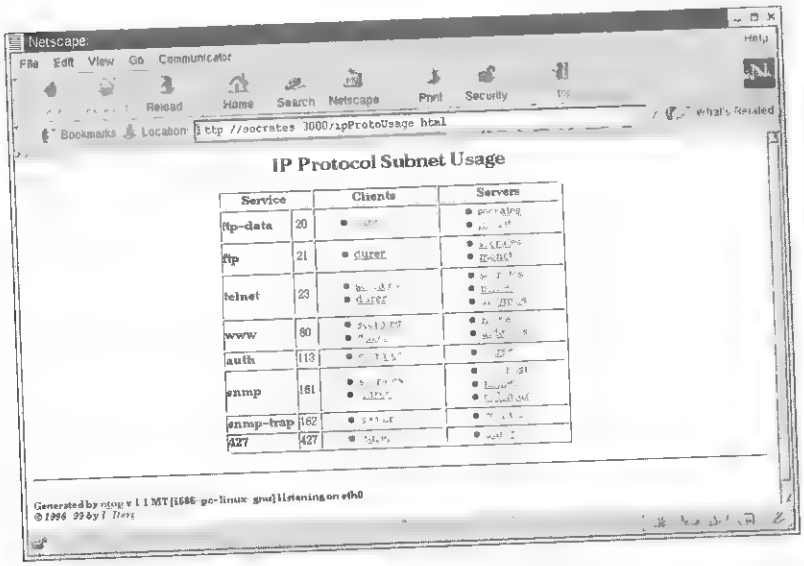


Figura 9.28 Reporte de subred del uso del protocolo.

Uso de las opciones de comando

La herramienta **ntop** soporta las opciones de línea de comandos incluidas en la tabla 9.10. Este software considera como un *host* inactivo a un *host* que no ha generado o recibido tráfico desde su último sondeo. Si se tiene interés en ver *hosts* inactivos y activos, se usa la opción `d*`. Al actualizarla mostrará ambos tipos cuando se generan reportes. Además, cuando **ntop** está corriendo en el modo terminal, la tecla **-d** se puede usar con el mismo propósito. Esto es útil para presentar todos los *hosts* sin importar si está activo o no. Otra opción práctica es `-i`, la cual controla la

N. del R.T.: En algunas plataformas, esta opción ejecutará a **ntop** como un demonio. Es decir, lo ejecutará en segundo plano.

interfaz de red que **ntop** usará para hacer seguimiento a la red. Con sistemas *multi-homed*, éste es un requerimiento absoluto.

Si es necesario hacer seguimiento de la red por porcentajes, se usa la opción `-p`. Esta opción presentará el tráfico como un porcentaje del tráfico total para cada sistema activo en la red.

Tabla 9.10 Opciones de línea de comandos.

Opción	Descripción
`-d`	Muestra todos los *hosts*, no sólo los activos
`-F`	Define un flujo (filtro) a lo que se le hace seguimiento y es reportado
`-l`	Muestra los *hosts* remotos y locales que generan tráfico de red
`-i`	Especifica una interfaz alterna que se puede usar
`-n`	Muestra direcciones IP numéricas en lugar de nombres
`-r`	Controla la demora entre actualizaciones de pantalla
`-p`	Muestra el tráfico de red como un porcentaje
`-w`	Habilita el modo Web

Cómo asegurar los reportes Web

Si se está preocupado por la seguridad respecto a los reportes, ¡no hay problema! La herramienta **ntop** soporta una característica de seguridad rudimentaria pero muy útil, que se puede implementar rápida y fácilmente. Cuando **ntop** está corriendo, éste busca un archivo llamado **.ntop** en el directorio de inicio del usuario que arranca el proceso. Este archivo puede contener uno o más nombres y contraseñas asociadas. Cuando se adiciona una entrada a este archivo, al usuario se le presenta una ventana de inicio de sesión, como se muestra en la figura 9.29.

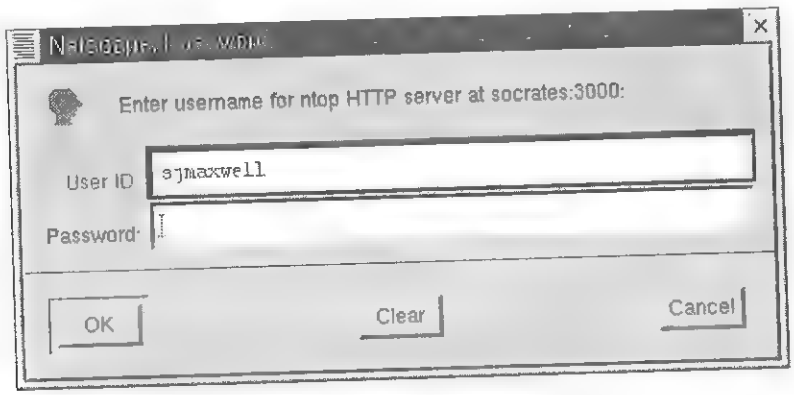


Figura 9.29 Ventana Ntop de inicio de sesión.

En la mayoría de los casos, es razonable que **ntop** se corra como el superusuario del sistema. Como resultado, se puede colocar un archivo `.ntop` en el directorio del sistema de archivos `/root`. Para hacer esto tan seguro como sea posible, cerciórese de que el archivo no sea legible, excepto por el superusuario. El permiso de `600` debería ser suficiente. Supóngase que se desea suministrar acceso a **ntop** para un puñado de usuarios. El siguiente ejemplo `.ntop` servirá para explicarlo:

```
netmgr private
tech-1 secret
smaxwell 1ghost99
acartwright 44pleaseread
```

Con estas entradas, cualquiera de los cuatro usuarios, `netmgr`, `tech-1`, `smaxwell` o `acartwright`, puede obtener acceso a los reportes **ntop**, siempre que cada uno de ellos use la cadena de contraseña asociada. Por tanto, `netmgr` tendría que usar `private`, y así sucesivamente. Esta característica de seguridad es muy simple, pero tiene su debilidad. Es decir, alguien podría capturar la secuencia de inicio de sesión con una sonda o con un software basado en el *host* y obtener el par nombre/contraseña.

Creación de flujos de seguimiento

La opción `-F` crea nuevos flujos en **ntop**. Sin embargo, la sintaxis es un poco difícil. El formato general al usar esta opción es el siguiente:

```
ntop -F "flow_name='filter expression'[,filter expression]"...
```

Cuando se use más de un flujo, cada uno debe ser seguido por el previo, separados por una coma sin ningún espacio enfrente de la coma. Toda la cadena después de la opción debe contener comillas dobles, mientras que la expresión del filtro está contenida en comillas sencillas. Supóngase que se necesitan dos filtros nuevos, uno para capturar todo el tráfico entre *hosts* `chips` y `dips` y el otro para capturar todo el trafico `tcp` desde el *host* `peanuts`. El primer flujo se llamará `chipsdips_traffic`, y el otro `tcp-peanuts`. El siguiente comando **ntop** con los flujos asociados se podría usar:

```
# ntop -F "chipsdips_traffic='nost chips and dips',tcp-peanuts='host\
peanuts and tcp'" -w 3000
```

En este comando, el primer flujo capturará todo el tráfico enviado o recibido por `chips` y `dips`. El otro flujo capturará todo el tráfico que contiene datos `tcp` y es enviado o recibido por `peanuts`. Recuérdese que **ntop** soporta muchas de las expresiones de filtro y comandos de palabra clave **ethereal** y **tcpdump**. Por tanto, se pueden presentar filtros poderosos para ayudar al seguimiento de la red.

Uso del modo terminal

Para quienes no quieran usar la interfaz Web, el segundo modo de presentación suministra salida a terminal similar al comando **top** UNIX. Para ello se invoca el comando **ntop** sin la opción `-w`. En este modo, la información de red se presenta en una ventana actualizada constantemente, como se muestra en la figura 9.30. La ventana contiene un estado dinámico de utilización de red ordenada por protocolos de red. Varias opciones de presentación están disponibles para alternar la información presentada y, aparte de eso, controlar el **ntop**. En la ventana que se muestra, los valores de tráfico de red se representan como un porcentaje. Esto fue habilitado por el comando **p**. La ventana sólo muestra algunos de los protocolos a los que se les hizo seguimiento; para desplazarse a través de toda la lista de protocolos se usa la barra espaciadora.

De manera predeterminada, **ntop** arranca en modo terminal. La información presentada con la interfaz Web está disponible en el modo terminal pero, por supuesto, los datos están formateados de forma diferente. Para revisar los diferentes reportes, **ntop** define un número de comandos de teclado que alterarán el reporte presentado. El comando de teclado consiste en presionar una sola tecla alfabética sin presionar la tecla Entrar.

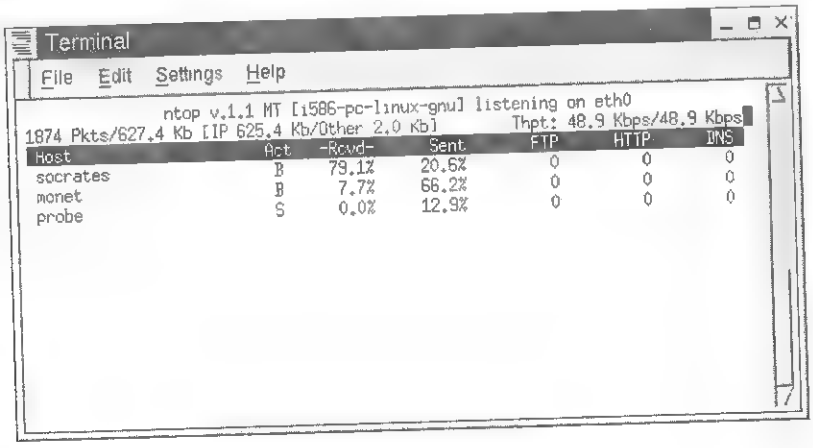


Figura 9.30 Presentación del modo terminal.

Por ejemplo, el comando **h** suministra ayuda para la herramienta **ntop** y presenta una lista de comandos de teclado disponibles. Por tanto, cuando **ntop** está corriendo, si se presiona esta tecla (sin presionar Retorno o Entrar) se presentará el menú de ayuda que se muestra en la figura 9.31.

Al usuario se le indica que debe presionar una tecla para retornar a la pantalla principal del reporte. Incluso cuando **ntop** está presentando la pantalla de ayuda, aún hace seguimiento y recolecta estadísticas de tráfico de red. Como se puede observar en la pantalla de ayuda, **ntop** muestra los comandos de teclado soportados. El comando **q** se usa para detener la operación de **ntop**. La secuencia de comandos **^c** llevará a cabo la misma función.

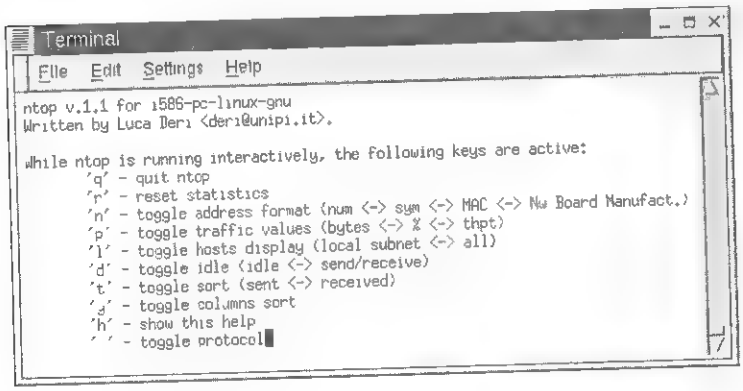


Figura 9.31 Modo terminal de ayuda.

10

Panel de control Linux

Una de las características importantes de Linux es que muchas labores y tareas administrativas de la red han sido automatizadas en una herramienta de configuración de interfaz gráfica de usuario (GUI) conocida como el panel de control. Este capítulo proporciona una vista general del panel de control suministrado con la versión estándar de Red Hat Linux. Quienes apenas empiezan a familiarizarse con UNIX y no quieren editar archivos especiales de configuración de red o usar comandos shell, encontrarán en el panel de control una buena ayuda. Aunque el panel de control no proporciona la capacidad para controlar cada aspecto del trabajo en red, sí suministra soporte para la mayoría de las tareas importantes.

El panel de control

El panel de control proporciona una GUI para varias tareas de configuración de red y de sistema. Puede usarse para hacer lo siguiente:

- controlar el encendido/apagado de los procesos o servicios
- cambiar la fecha y la hora
- configurar/controlar impresoras
- configurar los parámetros de red
- configurar los parámetros del módem
- cargar/descargar los módulos del núcleo (*kernel*)
- configurar parámetros adicionales del sistema

Esta herramienta se puede invocar desde la interfaz del modelo de objeto de entorno de red GNU (GNOME) a través del menú principal de la GUI o de la línea de comandos. Para iniciarlo desde la GUI, se da un clic una vez en el *triángulo con punta hacia arriba* GNOME, se navega hacia el menú `System`, luego se mueve a la derecha y se selecciona el submenú `Control Panel` (panel de control). La figura 10.1 muestra el menú principal GNOME y el menú `System`.

Como otra alternativa, el usuario puede tener acceso al panel de control desde la línea de comandos digitando lo siguiente:

```
# /usr/bin/control-panel
```

Se debería ser `root` cuando se corra el `panel de control`, porque éste modifica archivos estándar de configuración UNIX que requieren privilegios de administrador. A pesar de que el `panel de control` contiene varios iconos diferentes y suministra una cantidad significativa de funcionalidad, este texto se centra sólo en las áreas que se relacionan con el trabajo en red. Sin embargo, es importante proporcionar también una vista general de las otras funciones, haciendo énfasis en los elementos de la red, ya que pertenecen a la administración de la red. Se presenta primero una vista general básica de los servicios del `panel de control`, seguida de un repaso a fondo de las tareas relacionadas con el trabajo en red.

El componente de la red del `panel de control`, conocido como `configurador de red`, configura los servicios básicos de red. UNIX contiene una gran cantidad de archivos de configuración que almacenan información importante del sistema y del trabajo en red en uno o más formatos específicos. Afortunadamente, esta herramienta maneja la colocación de los datos en los archivos apropiados y en el formato correcto. Sin embargo, el administrador de UNIX necesitará saber dónde están localizados algunos de los archivos y sus formatos asociados, debido a que el acceso al panel de control puede no ser posible o poco práctico en ciertas situaciones.

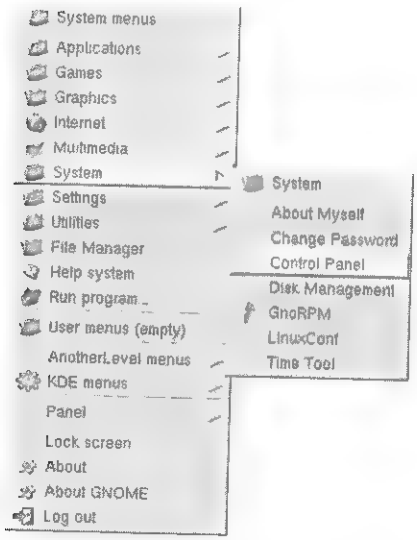


Figura 10.1 Acceso al panel de control.

Al iniciar el panel de control, presenta la ventana principal que se muestra en la figura 10.2. La ventana contiene un único menú, `File` (Archivo), más ocho iconos. El archivo `File` contiene dos elementos: `Change Orientation` (Cambio de la orientación) y `Exit` (Salir). La opción `Exit` se explica por sí sola, mientras que la opción `Change Orientation` actúa como un alternador y alterará la forma en que se presentan los iconos. En la figura 10.1, la orientación de los iconos es horizontal y el seleccionar el elemento `Change Orientation` hará que los iconos se presenten verticalmente. Al seleccionar la opción, retornará a la orientación previa.

Editor de nivel de ejecución

El primer icono da acceso al administrador del nivel de ejecución, el cual puede controlar los sistemas y/o los servicios de red que son activados en el arranque y los servicios que son desactivados cuando el sistema se apague. Linux, soporta el mecanismo estándar de *nivel de ejecución,* el cual tiene sus orígenes en las versiones UNIX AT&T, donde los servicios del sistema se invocan y detienen, basados en modos diferentes o niveles ejecución en el sistema. Por ejemplo, Linux normalmente opera en el `nivel de ejecución 5`, el cual se conoce como modo multiusuario. Éste es el modo operativo primario cuando el sistema se inicia y corre de manera normal. En otros sistemas como Solaris, el `nivel de ejecución 2` es el modo multiusuario sin los servicios NFS, mientras que el `nivel de ejecución 3` es el modo multiusuario normal. Un usuario puede determinar en qué nivel de ejecución está operando el sistema mediante el uso del comando Linux **runlevel**. También se puede usar el comando **uname -a**.

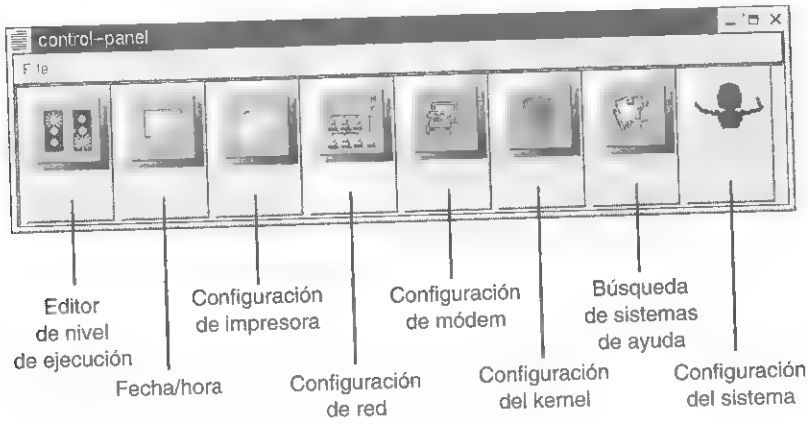


Figura 10.2 Ventana principal del panel de control.

Cambio de la fecha y la hora

El siguiente icono cambia la fecha y la hora del sistema. Se presenta una ventana de diálogo, la cual permite al usuario seleccionar componentes individuales (como hora, minuto y segundo) de la fecha/hora actual, de modo que se pueda cambiar.

Configuración/control de impresoras

Este icono presenta el administrador de impresión del sistema, el cual permite definir, controlar y borrar impresoras del sistema.

La herramienta soporta la creación de impresoras locales, remotas a través del demonio de impresora de línea (LPD), de Windows 95/NT y NetWare. También, permite que el usuario pueda controlar el proceso de impresión de segundo plano y el envío de páginas de prueba a las impresoras definidas como una ayuda de diagnóstico y depuración.

Configuración de la red

Este icono representa la herramienta configuradora de la red que controla los servicios básicos de ésta. La herramienta define propiedades del cliente: servidor de nombres de dominio, información de enrutado, interfaces del trabajo en red e información del *host*. La siguiente sección, "Uso del configurador de la red", suministra información más detallada.

Uso de los módulos del *kernel*

El sistema Linux soporta la carga dinámica de módulos del sistema. Los módulos suministran funcionalidad adicional o servicios al sistema, sin tener que construir un *kernel* nuevo o reiniciar el sistema. Este icono accede al `kernel configurator` (configurador del *kernel*) y permite la carga y descarga de módulos sobre la mancha. Por ejemplo, si el usuario quiere habilitar el soporte para una interfaz de red TokenRing, puede hacerlo cargando el módulo `tr`, que es uno de los módulos disponibles.

Acceso a la ayuda

El icono `help` (ayuda) se puede usar para configurar dónde buscará el sistema documentación en línea. Esto incluye, por ejemplo, la localización de páginas *man* (manuales).

Configuración del sistema

El icono final presenta la herramienta Linux de *configuración,* la cual configura y controla un gran número de parámetros del sistema y de red. El software de configuración también se puede arrancar desde la línea de comandos usando el comando **Linuxconf.** En la siguiente sección aparece una vista general de la herramienta.

Uso del configurador de la red

Como se mencionó previamente, el *configurador de la red* (`network configurator`) suministra la capacidad para configurar

parámetros básicos del trabajo en red. Las siguientes tareas se pueden llevar a cabo usando este software:

- configurar el parámetro de configuración del cliente DNS
- mantener la base de datos de nombres de *hosts*
- configurar, alterar o borrar interfaces de red
- configurar el enrutado de la red

Al seleccionar el icono del `configurador de red` del panel de control se presenta la ventana que muestra la figura 10.3.

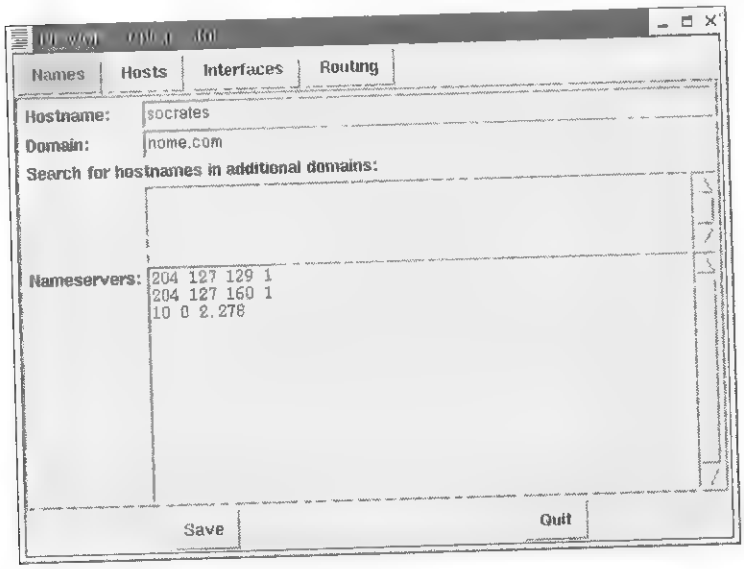


Figura 10.3 Ventana del configurador de red.

La parte superior de la ventana contiene un conjunto de botones que se usan para navegar en diferentes vistas o categorías. Cada botón representa una categoría. Se suministra un total de cuatro botones, y al dar clic en cualquiera de estos se redibujará la ventana principal, con la vista asociada.

Parámetros de configuración DNS

El primer botón, `Names` (nombres), es la vista predeterminada y siempre se presenta primero cada vez que se invoca el software de configuración. La vista se utiliza para definir el nombre del

host del sistema local y el nombre de dominio predeterminado. Además, el usuario puede configurar los servidores DNS para que consulten y busquen métricas. En la figura 10.3, el nombre del sistema, `socrates`, está contenido en el campo `Hostname` (nombre del anfitrión) y el nombre de dominio DNS, `home.com`, se lista en el campo `Domain` (dominio). Al cambiar el campo `Hostname`, cambiará el nombre del sistema. Si se quiere listar el nombre del sistema mientras no se usa el panel de control, se utiliza el comando **hostname**.

Los tres campos restantes, `Domain` (dominio), `Search for hostnames in additional domains` (buscar nombres de *host* en dominios adicionales) y `Nameservers` (servidores de nombre), controlan las operaciones del software cliente DNS. Los tres servidores de nombre definidos son `204.127.129.1`, `204.127.160.1` y `10.0.2.278`. Estas entradas dicen al sistema cuál servidor de nombre debería consultarse y en qué orden, cuando se usa la resolución del nombre del *host*. El campo de búsqueda dice al sistema cuáles dominios adicionales se deberían agregar al nombre del *host* cuando se consulta uno de los servidores de nombre definidos. Un servidor de nombre será responsable de soportar uno o más dominios a los cuales han sido asociados los *hosts* específicos. En este caso, `socrates` pertenece a un dominio `home.com`, pero el sistema también puede consultar *hosts* adicionales que pertenecen a otros dominios.

La información DNS se almacena en el archivo de configuración `/etc/resolv.conf`, y un cambio del nombre del *host* actualiza automáticamente estos archivos. Nótese que cualquier información previa guardada en los archivos de configuración será sobreescrita con los contenidos de los campos cada vez que se guardan.

Cómo mantener la base de datos de *host*

El botón `Hosts` (anfitriones) mantiene entradas con el archivo `/etc/hosts`; al seleccionar este botón se presentan los contenidos de este archivo para la vista actual.

Nota

Los cambios se hacen sólo al archivo local del *host*. Si se está corriendo NIS o DNS, los cambios hechos en estos archivos no afectarán a ninguno de estos servicios de resolución.

Este archivo del sistema contiene transformaciones de dirección de nombre de *host* a IP, y consta hasta de tres campos: `IP`, `Names` y `Nicknames`. La dirección IP representa la dirección única para el *host* designado. Esta dirección es algo no se obtiene por arte de magia. Esta dirección es asignada específicamente a un sistema o, en el caso de usar asignación dinámica de IP, es asignada por un servidor al *host* específico, quien la obtiene cuando éste inicia. Normalmente, si se está corriendo NIS o DNS, no se necesitará este archivo. Lo más importante es que todas las direcciones IP asignadas a una red deben ser únicas para que los protocolos TCP/IP funcionen correctamente.

La figura 10.4 muestra un ejemplo de conjunto de nombres de *hosts* ya definidos en el sistema local. Estas entradas representan sólo transformaciones entre las direcciones IP del nodo y alguna designación del nombre de *host*. El nombre del *host* es sólo para el "consumo" humano. Los nombres simplemente facilitan que las personas distingan un sistema del otro y se desentiendan de la operación de los protocolos TCP/IP. El software de trabajo en red usa direcciones, no nombres. Mientras que las direcciones IP son un conjunto fijo de números, los nombres de *host* son un poco más amplios. Un *host* puede llamarse de cualquier manera y, como se muestra en la figura 10.4, los nombres pueden variar en longitud e incluir números y la mayoría de caracteres válidos.

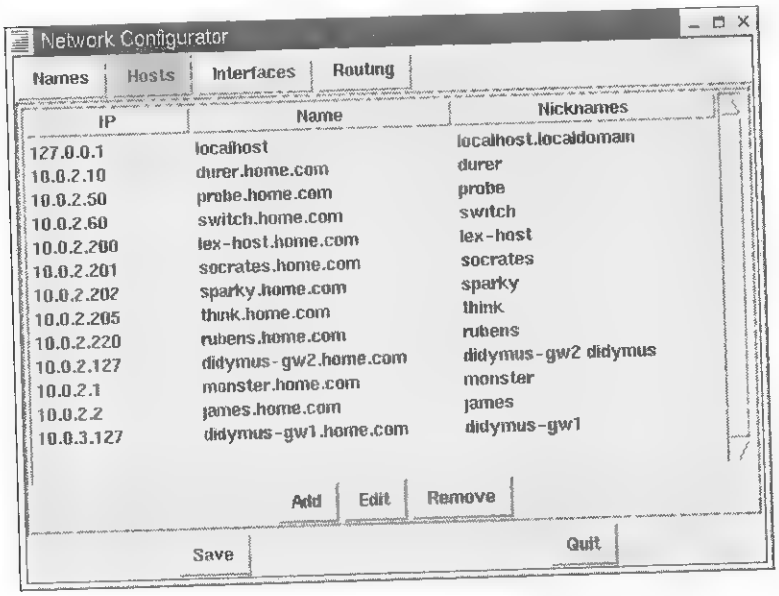


Figura 10.4 `Configurador de red`, vista de *hosts*.

Como se puede observar, la ventana vista de *host* define tres botones, `Add` (agregar), `Edit` (editar) y `Remove` (remover), que ayudan al usuario a definir, cambiar y remover entradas específicas de *host*. La vista divide cada línea en tres campos, que corresponden al formato del archivo `/etc/hosts`. El formato incluye dirección IP, nombre oficial de *host* y alias. Para agregar un *host* nuevo, se da un clic en el botón `Add`, y aparecerá una pequeña ventana de diálogo, la cual tiene tres campos en blanco: `IP`, `Name` (nombre) y `Nickname` (apodo). La figura 10.5 muestra esta ventana.

Supóngase que se desea agregar el *host* `monet` que tiene una dirección IP correspondiente de `10.0.2.100`; para ello se agregarían estas cadenas en los campos apropiados en cada uno de los cuadros de texto. Desafortunadamente, el software del panel de control no hace ninguna verificación de error en el formato o en la validez de la información especificada en estos campos, lo que significa, por ejemplo, que el usuario puede introducir datos incorrectos o aun mal formateados y que el software los aceptará sin ninguna advertencia.

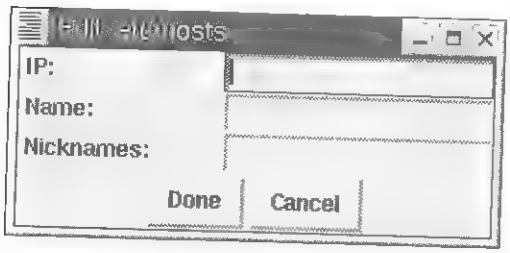


Figura 10.5 Ventana de diálogo de *host* agregado.

En la práctica, la mayoría –si no todos– de los archivos de configuración UNIX (incluidos `/etc/hosts`) normalmente no tienen ningún mecanismo integrado para asegurar que la información introducida es correcta. Muchos archivos, sin embargo, suministran algún tipo de verificación de error para señalar errores específicos relacionados con formato o sintaxis. Esto por lo general se lleva a cabo corriendo el programa asociado que es responsable de analizar el archivo específico de configuración y operar en sus contenidos para suministrar alguna aplicación específica o servicio del sistema. Sólo en este punto se detecta cualquier error. El software del panel de control suministra una verificación rudimentaria; para que una entrada válida sea aceptada, deben

llenarse los campos primero y segundo, mientras que el tercero es opcional y puede dejarse en blanco.

Una vez introducida la información deseada, el usuario debe dar clic en el botón `Done` (hecho). Esta acción agregará la entrada a la lista de transformaciones existentes de *host*. Sin embargo, el usuario debe guardar la nueva configuración seleccionando el botón `Save` (guardar). Si el usuario no guarda la nueva entrada y sale del `panel de control`, la nueva entrada se perderá. Desafortunadamente, el software no proporciona una advertencia si el usuario decide salir del software antes de guardar la nueva configuración del *host*.

Para alterar una entrada existente de *host*, se selecciona desde la ventana principal y se da un clic en el botón `Edit` (editar). Se mostrará la misma ventana que se muestra en la figura 10.5, pero ésta incluirá ya la información del *host*. A continuación se cambia la información deseada y se da un clic en el botón `Done` (hecho). Si un anfitrión debe borrarse de la lista, simplemente se selecciona y se da un clic en el botón `Remove` (remover).

Si se quiere estar seguro de que la entrada se agregó correctamente, sólo se usa el comando **grep** y se busca el nombre de *host* nuevo desde el archivo `/etc/hosts`. Por tanto,

```
# grep monet /etc/hosts
```

retornará lo siguiente, si el sofware del *panel de control* agregó correctamente la entrada del *host*:

```
10.0.2.100 monet
```

Configuración de interfaces de red

El botón `Interfaces` agrega, edita o remueve interfaces de red del sistema. Al seleccionar el botón `Interfaces` se presentan las interfaces actuales definidas en la ventana que se muestra en la figura 10.6. Esta ventana muestra la información de configuración de la interfaz en campos separados y contiene algunos de los mismos botones de la vista de *hosts* ya analizada. Sin embargo, la vista `interfaces` tiene algunos botones extra: `Clone` (clon), `Alias` (alias), `Activate` (activar) y `Deactivate` (desactivar).

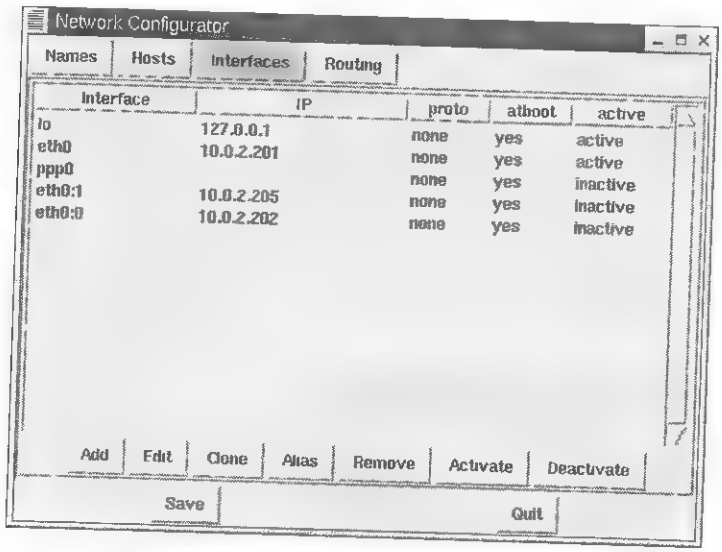


Figura 10.6 Configurador de red, vista de la interfaz.

Para agregar una interfaz al sistema, el usuario debe dar clic en el botón `Add` y seleccionar el tipo desde la ventana de diálogo de selección de la interfaz, que se muestra en la figura 10.7. El usuario debe seleccionar la interfaz específica que se va a configurar. Las primeras tres interfaces, más la última, `PPP`, `SLIP`, `PLIP` y `Pocket`, se usan con cualquier puerto, serial o paralelo, que está disponible en la mayoría de los sistemas de computadora. Las interfaces restantes, `Ethernet`, `Arcnet` y `TokenRing`, requieren una interfaz de hardware específica que debe adicionarse al sistema antes de poder usar la interfaz. Por ejemplo, si el usuario deseara configurar una segunda interfaz `Ethernet`, la segunda tarjeta de interfaz debería estar ya instalada en el sistema.

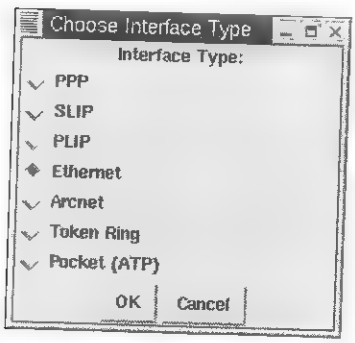


Figura 10.7 Ventana de selección de la interfaz.

Debe tenerse en cuenta que el usuario puede agregar varias interfaces usando el panel de control, pero sólo el hardware instalado determinará si esas interfaces se pueden usar realmente.

Dependiendo del tipo de interfaz que se escoja en la ventana de selección de la interfaz, se presentará un conjunto diferente de preguntas en una nueva ventana. Esto se debe a que algunas de las interfaces requieren información diferente antes de que se puedan configurar. Por ejemplo, al seleccionar el tipo Ethernet y pulsar el botón `OK`, se presenta la ventana que se muestra en la figura 10.8.

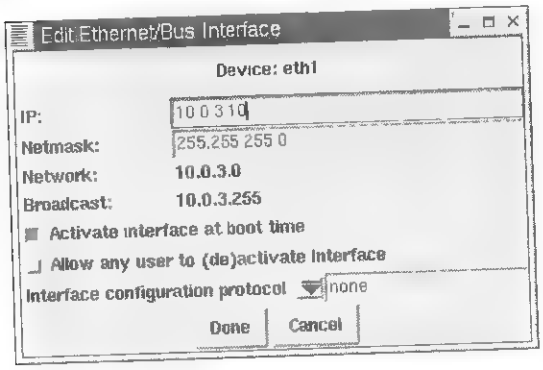


Figura 10.8 Ventana de interfaz Ethernet.

Cuando se agrega una interfaz Ethernet nueva, se deben completar los campos `IP` y `netmask`. Si se agrega una dirección IP válida al campo `IP`, los campos `Netmask`, `Network` y `Broadcast` se llenarán automáticamente. El campo `Netmask` representa la máscara de subred de 32 bits que se usará con la dirección IP para determinar el número de subred. El valor para el campo `Netmask` lo determina la máscara predeterminada de la dirección de clase `IP` introducida en el campo IP. Si este valor no es apropiado, se cambia el `Netmask` para que se adecue a los requerimientos del trabajo en red.

Se debe revisar el botón `Activate interface at boot time` (activar la interfaz en el momento de inicio) si la interfaz se tiene que arrancar cuando el sistema se inicia. En la mayoría de los casos, la interfaz debería estar habilitada en el arranque, de modo que los protocolos de nivel superior o servicios de red estén disponibles. Se supone que el botón `Allow any user to (de)activate interface` (permitir a cualquier usuario {des}activar la interfaz) controla si los usuarios diferentes a root pueden habilitar o inhabilitar esta interfaz. Desafortunadamente,

esta característica no parece funcionar. El menú desplegable y/o cuadro de texto `Interface configuration protocol` (protocolo de configuración de interfaz) especifica el método que se debería usar para obtener datos adicionales de la configuración de la interfaz o información de la dirección IP. Los dos posibles métodos son el protocolo de configuración dinámica del *host* (`DHCP`) y el protocolo de arranque (`BOOTP`) que suministran un mecanismo para recuperar información de dirección IP y otra información para un servidor central. Ambos requieren uno o más servidores que estén operando en la red.

Una vez configurada la interfaz, se da un clic en el botón `Done`. A continuación, otra ventana de diálogo indicará al usuario que guarde los cambios de la interfaz. La figura 10.9 muestra esta ventana de diálogo. Al dar clic en el botón `Save` se guardará permanentemente la configuración, mientras que el botón `Cancel` (cancelar) no guardará la información. Si se da clic en el botón `Cancel` de la ventana de edición de la interfaz, se borrará el cambio que se hizo.

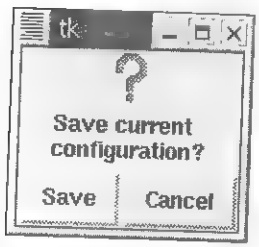


Figura 10.9 Ventana de diálogo *Save*.

Cuando se adiciona una nueva interfaz, se coloca un archivo de configuración en el directorio `/etc/sysconfig/network-scripts`. El nombre del archivo contiene el prefijo `ifcfg`, seguido del nombre de la interfaz. Cuando se crea una interfaz nueva llamada `eth1`, se obtiene un archivo de configuración llamado `ifcfg-eth1`. Este archivo contiene la información de la interfaz que se muestra a continuación:

```
DEVICE=eth1
IPADDR=10.0.2.220
NETMASK=255.255.255.0
NETWORK=10.0.2.0
BROADCAST=10.0.2.255
ONBOOT=yes
BOOTPROTO=none
USERCTL=no
```

Como se puede observar en la salida, el archivo contiene una lista de nombres seguidos del signo igual, más un valor asociado. La razón principal para incluir esta información es que si se tiene que hacer, se podrá realizar un cambio rápido en la secuencia de comandos sin la ayuda del panel de control GUI. Cuando se modifica una interfaz, este archivo simplemente se actualiza con la nueva información; cuando una interfaz se borra, se remueve el archivo.

Si el tipo de interfaz (`ppp`) punto a punto (PPP) se selecciona en la ventana que se muestra en la figura 10.7, entonces se presentará una ventana de diálogo de configuración diferente. Esta ventana, que se muestra en la figura 10.10, contiene los parámetros de configuración PPP requeridos.

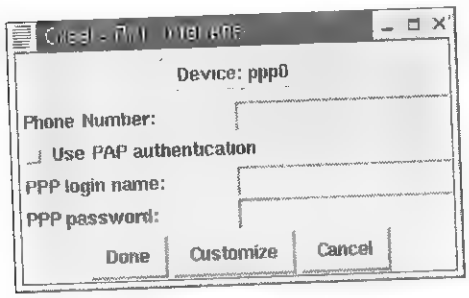


Figura 10.10 Ventana de diálogo de interfaz PPP.

Cuando se usa PPP, se necesitan varias piezas importantes de información para asegurarse de que se pueda establecer un enlace exitoso con el extremo remoto. Lo primero y más obvio es que se debe tener una cuenta, con un proveedor de servicios de Internet (ISP), si la conexión se va a usar para suministrar conectividad a la Internet. Si éste no es el caso, se debe contar con otro sistema que tenga un módem que pueda marcar. El número de teléfono del sistema remoto se debe colocar en el campo `Phone Number` (número de teléfono). A continuación, se debe tener una cuenta con un nombre de usuario y una contraseña. Si se va a usar el protocolo PAP, se debe dar clic en el elemento `Use PAP authentication` (use la autenticación PAP). Después, se debe introducir el ID del usuario (login) y la contraseña asociada en los campos `PPP login name` (nombre de inicio de sesión PPP) y en `PPP password` (contraseña PPP), respectivamente. El paso final del proceso de configuración exige un entendimiento de los detalles de la conexión: opciones del puerto del sistema, velocidad y otros

parámetros. Si se da clic en el botón `Done`, se usarán los valores predeterminados que pueden no ser compatibles con su entorno. Se recomienda dar clic en el botón `Customize` (personalización), para asegurar y/o corregir los valores predeterminados no apropiados para el sitio. Al dar clic en el botón `Done` se presenta una nueva ventana que tiene cuatro marbetes de panel: `Hardware` (equipo), `Communication` (comunicación), `Networking` (trabajo en red) y `PAP`. La figura 10.11 muestra esta ventana.

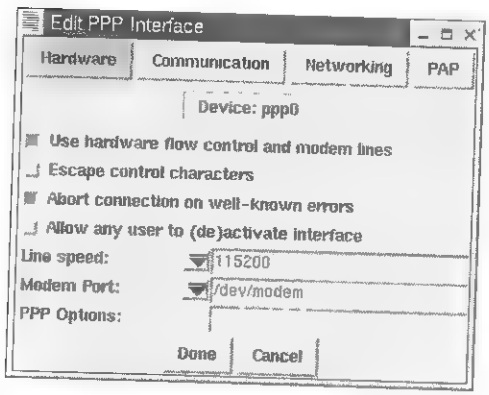


Figura 10.11 Ventana de personalización PPP.

El panel `Hardware` (equipo), el cual contiene el nombre de la interfaz que se está configurando, se usa para configurar algunos de los parámetros operativos de capa inferior necesarios. En este caso, la interfaz es `ppp0`. La primera casilla de verificación, `Use hardware flow control and modem lines` (use control de flujo de hardware y líneas de módem), habilitará o inhabilitará el control del flujo para el módem y el puerto asociado que conecta el módem. El control del flujo asegura que cuando se establece una conexión, tanto los módems como los puertos puedan controlar el flujo de información sin perder ningún dato debido a un error. Esta casilla de verificación debería mantenerse marcada en la mayoría de las situaciones de conectividad remota. La opción `Escape control characters` (caracteres de control de escape) se debería inhabilitar, porque se usa para asegurarse de que el punto local o módem no interpretará los caracteres de control enviados. La opción `Abort connection on well-known errors` (abortar la conexión en errores bien conocidos) sólo significa que si ocurren uno o más errores comunes, la conexión se terminará. Esta opción debería ser habilitada porque es una buena idea romper la

conexión cuando ocurre un problema en la capa inferior. El botón `Allow any user to (de)activate interface` (permitir a cualquier usuario [des]activar la interfaz) se suministra para ayudar a los usuarios no privilegiados a que habiliten la interfaz. Esto es útil, por ejemplo, cuando ocurre un problema temporal en el extremo remoto y el administrador de red no está cerca para restablecer la conexión cuando los servicios se restauran y un usuario regular debe ayudar a esta tarea.

Los dos cuadros de texto siguientes contienen parámetros específicos para el puerto de hardware que se está usando. El campo `Line Speed` (velocidad de línea) controla la tasa de baudios para la interfaz física. Éste se debe establecer en la tasa más alta posible que la interfaz pueda soportar, teniendo en cuenta que, independientemente del valor en este campo, la tasa real del puerto estará limitada a lo que suministre el hardware. Por ejemplo, si el puerto sólo soporta `57600` baudios, pero la velocidad de línea se configura para `116200`, se usará la tasa `57600` sin importar el valor establecido de la configuración. A continuación, el campo `Modem Port` (puerto del módem) debería contener el nombre del puerto físico al cual está conectado el módem. De manera predeterminada, se realiza un enlace simbólico entre el puerto real y `/dev/modem`. Es mejor usar el enlace simbólico en esta ventana de configuración, porque el puerto se puede cambiar posteriormente, y el puerto `/dev/modem` se puede reenlazar con otro puerto sin cambiar la configuración. Si se necesita cualquier opción de configuración especial punto a punto, se puede agregar al campo `PPP Options` (opciones PPP).

El panel `Communication` (comunicación), que se muestra en la figura 10.12, contiene comandos de módem y parámetros adicionales como las cadenas espera/respuesta. El primer campo, `Modem Init String` (cadena de inicio del módem), reinicia el módem antes de que la interfaz se active. Siempre es una buena idea hacer esto porque asegura que el sistema puede hablar al módem y que el módem sea iniciado a un estado operativo conocido. El campo `Modem Dial Command` (comando de marcado del módem) almacena la secuencia de comandos para marcar el número de teléfono localizado en el campo `Phone Number` (número telefónico). Si se experimentan problemas para conseguir que una conexión funcione correctamente, se marca la opción `Debug Connection` (conexión de depuración). Esto generará información adicional de depuración que se agrega al archivo `/var/adm/messages`. La porción restante de la ventana define una secuencia de comandos `expect` (esperados) que se usa para registrarse

en el extremo remoto. Algunos sistemas presentan series de indicadores de comandos de inicio de sesión y exigen respuesta a éstos en un orden particular. La secuencia de comandos `expect` lee estos indicadores de comandos y emite la respuesta correcta al sistema remoto. Sin embargo, cuando se usa el protocolo PAP, estas secuencias de comandos no deben ser necesarias.

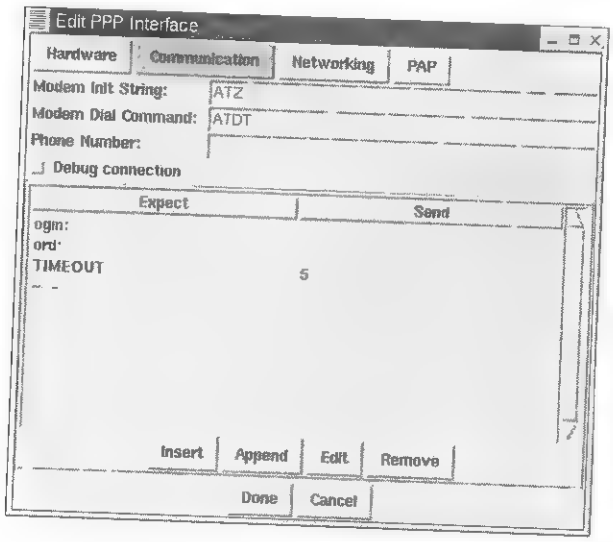


Figura 10.12 Ventana de marbete Comunicación.

El panel de `Networking` (red), que se muestra en la figura 10.13, suministra acceso a los parámetros de la capa de red y a las opciones. La opción `Activate interface at boot time` (activar interfaz en el momento de arranque) debería marcarse si se debe iniciar la interfaz cuando el sistema arranca. Esta característica se usaría, por ejemplo, cuando el sistema se utiliza como una ruta, servidor o estación de trabajo que requiere una conexión remota cuando el sistema se activa, no sólo cuando los usuarios se registran en el sistema. Cuando se selecciona esta opción, la bandera `ONBOOT` se establece en `yes` (en el archivo de configuración de la interfaz); cuando el sistema inicia, la secuencia de comandos de inicio `/etc/rc.d/rc2.d/S10network` examina esta bandera e inicia cualquier interfaz que haya habilitado la bandera `ONBOOT`.

La opción `Set default route when making connection` (establecer ruta predefinida cuando se hace conexión) configurará la tabla de enrutado local y adicionará una ruta predeterminada a la red asociada con esta interfaz. Esta opción está habilitada de

manera predeterminada. Es muy útil en situaciones en las que esta interfaz representa conectividad a la Internet y a la mayoría de tráfico de red en esta interfaz. Estableciendo la ruta predeterminada, se elimina la necesidad de correr un protocolo de enrutado de red como RIP u OSPF. En ciertas situaciones, es deseable correr uno de estos protocolos. En muchos casos, sin embargo, el enfoque de la ruta predefinida será suficiente, particularmente en casos en que la estación de trabajo contiene sólo una interfaz de red hacia el mundo exterior.

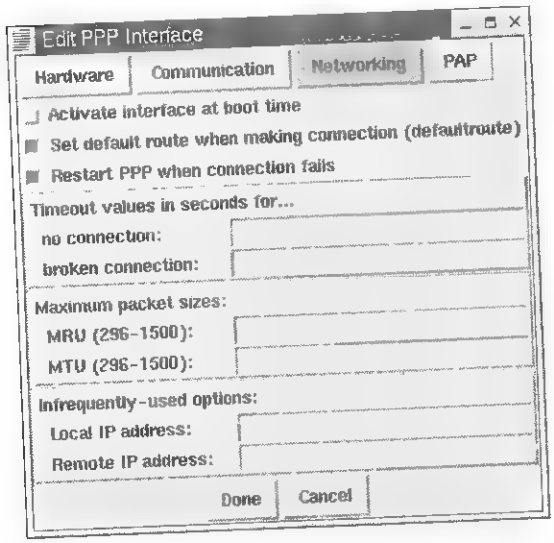


Figura 10.13 Panel de trabajo en red.

La opción `Restart PPP when connection fails` (rearrancar PPP cuando la conexión falla) también está habilitada de manera predeterminada y tiene mucho sentido reiniciar el servicio PPP si una conexión falla. Debido a que las fallas en la conexión pueden ser causadas por un gran número de problemas, y algunos entornos quieren asegurar la conectividad cada vez que sucede un problema, la conectividad al sitio remoto se reiniciará automáticamente. Por supuesto, si ocurre un problema en particular, por ejemplo, que el extremo remoto se caiga, reiniciar la conexión una y otra vez no servirá de nada. Sin embargo, cuando un problema con el extremo remoto se resuelve finalmente, ninguna intervención humana es necesaria para establecer la conexión en este punto. Puede llegar a ser necesario limitar la cantidad de tiempo gastado intentando establecer la conexión remota.

Los siguientes dos cuadros de texto contienen valores de extemporaneidad (en segundos) asociados con la interfaz cuando ocurre una falla en el enlace. El campo `no connection` (no conexión) controla cuánto tiempo debería transcurrir cuando un estado de no conexión se detecta en la interfaz. El campo `broken connection` (conexión rota) controla la cantidad de tiempo entre intentos de conexión cuando el enlace se termina, ya sea por el extremo remoto o debido a otro problema (por ejemplo, problemas en la línea telefónica). La segunda sección contiene los máximos del tamaño del paquete, los cuales, en circunstancias normales, no es necesario alterar. La sección final define direcciones IP estáticas, tanto para las interfaces de red locales como remotas. En la mayoría de las ISP, la información de la dirección IP se asigna dinámicamente y, como resultado, no se debería incluir información de dirección alguna.

El panel `PAP` almacena el nombre del usuario y la contraseña para el inicio de sesión PAP. La figura 10.14 muestra este panel, el cual contiene una sola entrada de inscripción. En este ejemplo, el nombre del usuario `2345678@worldnet.att.net` tiene la contraseña asociada `abcdefg-zxcv`. El nombre del usuario tiene formato de acuerdo con las convenciones establecidas por el ISP quien normalmente implementa sus propios estándares para nombres de inicio de sesión.

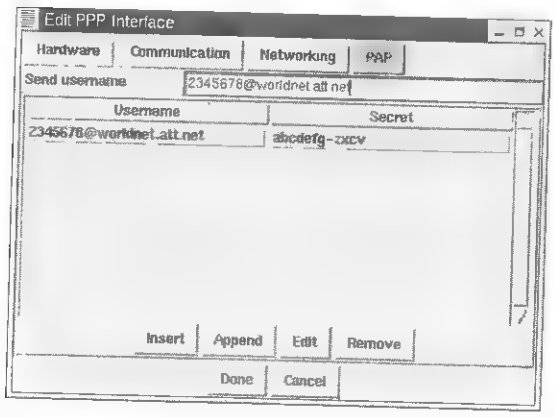


Figura 10.14 Panel de configuración PAP.

El panel `interfaces` también contiene botones adicionales que mantienen y controlan las interfaces existentes. El botón `Edit` modifica los parámetros de una interfaz existente. Si se

quiere editar una interfaz, primero se selecciona con el ratón, luego se da clic en el botón `Edit`. Después de eso, se presentará una ventana, como se mostró previamente en la figura 10.8. El siguiente botón, `Clone`, agrega una nueva interfaz usando los parámetros y las opciones de otra interfaz. Al dar clic en este botón, se presenta la ventana de diálogo que se muestra en la figura 10.15, luego se introduce el nombre de la nueva interfaz que será una copia (clon) de la interfaz seleccionada.

Al dar clic en el botón `Done`, se creará un alias de una de las interfaces existentes. Un alias de interfaz es igual que una interfaz lógica o una pseudointerfaz. En pocas palabras, un alias define otra interfaz que es una copia de una interfaz existente pero que usa una dirección de red diferente. El propósito principal de un alias de interfaz es crear otra dirección de red para la máquina. El alias contiene todos los parámetros y las características operativas de la interfaz original.

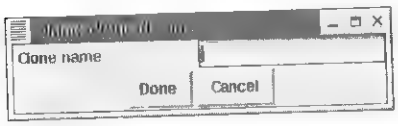


Figura 10.15 Ventana de clon.

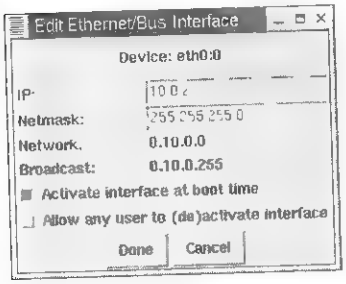


Figura 10.16 Ventana de alias de interfaz.

Al seleccionar el botón `Done` se presenta la ventana `Edit Ethernet/Bus interface` (Editar Ethernet/interfaz bus) que se muestra en la figura 10.16. Ésta es la misma ventana que se presenta cuando se edita una interfaz existente. Sin embargo, el sistema agrega un número de alias después del nombre de la interfaz, separado por un carácter de dos puntos. En la figura 10.16, se agregó un cero al nombre `eth0`. Así es como el sistema especifica los alias de las interfaces. En este caso, el primer alias

para la interfaz Ethernet se llama `eth0:0`. Si se crea otro alias, será etiquetado como `etn0:1`, el tercero como `eth0:2`, y así sucesivamente. Los alias de interfaces actúan y aparecen en el sistema como cualquier interfaz normal, y los comandos de trabajo en red como **ifconfig** también se pueden usar. Por ejemplo, considérese la salida del comando **ifconfig -a**:

```
eth0      Link encap:Ethernet HWaddr 00:10:5A:28:5D:7C
          inet addr:10.0.2.201 Bcast:10.0.2.255 Mask:255.255.255.0
          UP BROADCAST RUNNING PROMISC MULTICAST MTU:1500 Metric:1
          RX packets:1494 errors:0 dropped:0 overruns:0 frame:0
          TX packets:842 errors:0 dropped:0 overruns:0 carrier:40
          collisions:0 txqueuelen:100
          Interrupt:10 Base address:0xfc00

eth0:0    Link encap:Ethernet HWaddr 00:10:5A:28:5D:7C
          inet addr:10.0.2.254 Bcast:10.0.2.255 Mask:255.255.255.0
          UP BROADCAST RUNNING PROMISC MULTICAST MTU:1500 Metric:1
          Interrupt:10 Base address:0xfc00

lo        Link encap:Local Loopback
          inet addr:127.0.0.1 Mask:255.0.0.0
          UP LOOPBACK RUNNING MTU:3924 Metric:1
          RX packets:7890 errors:0 dropped:0 overruns:0 frame:0
          TX packets:7890 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
```

La salida incluye tres interfaces: `eth0`, `eth0:0` y `lo`. Como se puede observar, se define una única pseudointerfaz llamada `eth0:0`. Nótese que tiene todas las características de la interfaz `eth0`, pero la dirección IP es diferente. Se puede encontrar información adicional acerca de los alias de interfaces en el capítulo 4, "Herramientas y utilidades centrales del sistema", en la sección **ifconfig**.

El botón `Remove` borra una interfaz existente definida en el sistema. Al dar clic en el botón `Remove` se presenta una ventana de diálogo, mostrada en la figura 10.17, que verificará que la interfaz especificada ha sido removida. Nótese que una vez que al botón `OK` se le ha dado clic, la interfaz se remueve permanentemente.

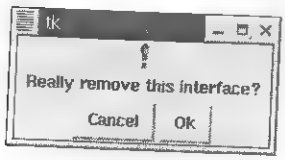


Figura 10.17 Ventana de verificación de remoción de la interfaz.

Como resultado, cualquier parámetro específico de configuración relacionado con la interfaz que se muestra en la figura se perderá para siempre. Se podría haber interés en guardar una copia del archivo de configuración relacionado con esta interfaz, antes de removerla, para cuando se necesite en el futuro. En este caso, se busca en el directorio `/etc/sysconfig/network scripts` un archivo con un nombre igual al nombre de la interfaz que se debería guardar.

Configuración de enrutado

El panel `routing` (enrutado) configura la puerta de enlace predeterminada y adiciona rutas estáticas a la tabla de enrutado. La opción `Network Packet Forwarding (IPv4,` transmisión del paquete de red [IPv4]) se suministra para activar el enrutado entre las redes configuradas en el sistema. Si el sistema contiene por lo menos dos interfaces y el tráfico IP necesita ser enrutado entre estas interfaces, debe marcarse esta opción; de otra manera, debería dejarse sin marcar. Realmente, la elección de enrutar implica más que simplemente habilitar esta opción. La selección y configuración del protocolo de enrutado también es necesaria, pero está más allá del alcance de este libro. De modo predeterminado, esta característica está inhabilitada. La figura 10.18 muestra el panel de enrutado (`routing`).

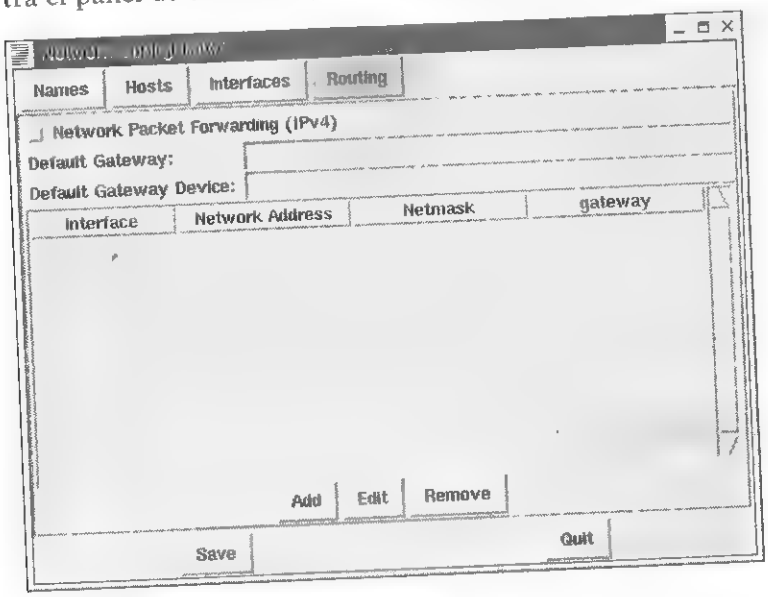


Figura 10.18 Panel de enrutado.

Los dos campos siguientes son usados por la puerta de enlace predeterminada. La *puerta de enlace predeterminada* (o enrutador) se usa cuando un protocolo de enrutado como `RIP` no está instalado o configurado en un sistema. Usar un enrutador predeterminado significa que si se debe tener acceso a cualquier red que no esté explícitamente definida en la tabla de enrutado, se usa el camino de enrutado predeterminado. Se introduce la dirección IP o el nombre de *host* válido del enrutador predeterminado en el campo `Default Gateway`. El campo `Default Gateway Device` (dispositivo predeterminado de puerta de enlace) contiene el nombre de la interfaz que se usa para tener acceso al enrutador predeterminado. Esta información se almacena en el archivo `/etc/sysconfig/network`. La porción restante del panel define entradas adicionales de enrutado estático. Normalmente, las rutas estáticas son necesarias cuando el sistema no está corriendo un protocolo de enrutado. Para agregar una ruta, se da un clic al botón `Add`. Este presentará un cuadro de diálogo que tiene cuatro campos, como se muestra en la figura 10.19. Una ruta estática requiere cuatro elementos: la interfaz de red de la ruta, la red enrutada, la máscara de red y el dispositivo de puerta de enlace/enrutador.

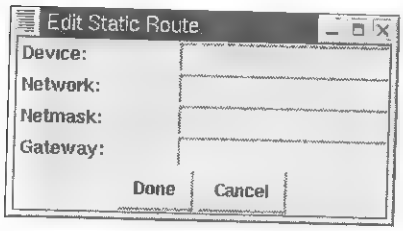


Figura 10.19 Entrada de ruta estática.

El campo `Device` (dispositivo) contiene el nombre de la interfaz de red que se usará con el fin de tener acceso a la red para la ruta. El campo `Network` (red) contiene la dirección IP punteada decimal o el nombre de red válido de la red a la cual pertenece la ruta. El campo `Netmask` (máscara de red) determina la porción de red y de nodo, particularmente si se implementa la subred. El campo `Gateway` (puerta de enlace) es la dirección IP real de la puerta de enlace o del dispositivo enrutador que reenviará paquetes a la red definida en el campo `Network`. Al dar clic en el botón `Done` se guardará la entrada de ruta en el archivo `/etc/sysconfig/static-routes`. Nótese que si hubiera rutas previamente definidas, aparecerían en la ventana que se presentó en la figura 10.14. Como una alternativa, el comando **route** se

puede usar para listar y adicionar entradas estáticas de enrutado. Desafortunadamente, cualquier entrada agregada con este comando no es guardada a través de reinicios del sistema. Consúltese en la página del manual Linux el comando **route**, para información adicional. Además, si se quiere listar la tabla de enrutado, se puede usar el comando **netstat** con la opción `-r` (para enrutado). El capítulo 4, "Herramientas y utilidades centrales del sistema", describe completamente este comando.

Uso del configurador del sistema

El panel de control proporciona acceso a otra herramienta del sistema, llamada `gnome-linuxconf`. Esta herramienta suministra un *front end* GUI para más tareas de administración de la configuración de red. Aunque esta herramienta no se describe completamente aquí, para concluir, se proporciona una vista general. Al seleccionar el icono de configuración del sistema del panel de control GUI se presenta la ventana que se muestra en la figura 10.20.

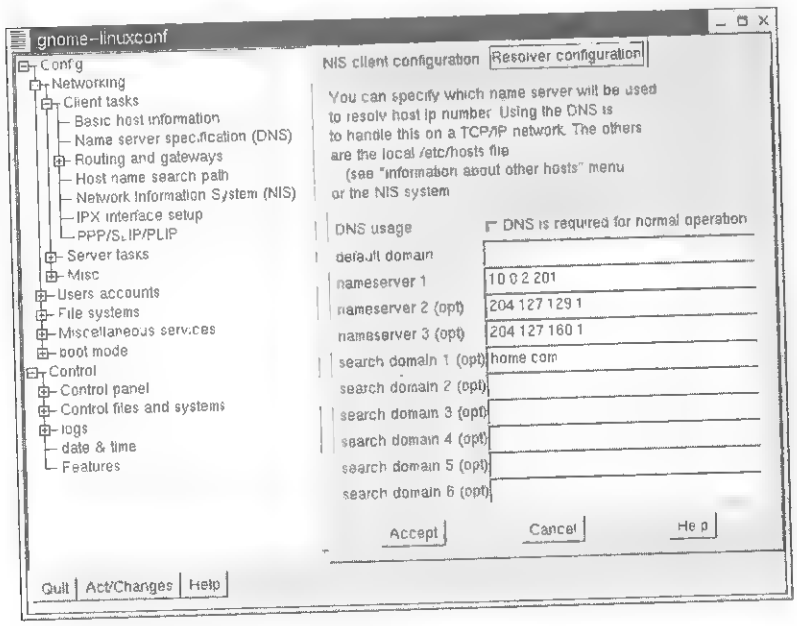


Figura 10.20 Ventana de configuración Linux.

La ventana `gnome-linuxconf` contiene una vista jerárquica de árbol a la izquierda, con uno o más paneles membreteados a la derecha. La vista de árbol contiene dos categorías raíz separadas, etiquetadas `Config` y `Control`, con un cuadro de estado asociado en la izquierda de cada etiqueta de categoría. Al dar un solo clic en el cuadro de estado, se expanden o contraen los elementos adicionales debajo de la categoría. Por ejemplo, dando un clic en el cuadro de estado de la etiqueta `Networking` (trabajo en red) se presentarán elementos adicionales. Al dar otra vez clic en la misma categoría, desaparecerán los elementos. Cada vez que el usuario selecciona una categoría, se presenta un nuevo panel membreteado. La figura 10.20 muestra dos paneles: `NIS client configuration` (configuración del cliente NIS) y `Resolver configuration` (configuración para resolver). Puesto que los paneles membreteados son acumulativos, es decir, cada categoría seleccionada presenta otro panel, el usuario debe borrar explícitamente un panel dando un clic en el botón `Quit` (renunciar) o `Close` (cerrar) asociado con ese panel.

Para ayudar al lector a que aprenda más sobre todas las funciones suministradas por la herramienta de configuración `gnome-linuxconf`, las tablas 10.1 y 10.2 listan y describen cada función de primer orden.

La categoría raíz `Config` contiene varias subcategorías para configurar las aplicaciones del servidor de trabajo en red:

- `Trabajo en red (Networking)`
- `Cuentas de usuarios (Users Accounts)`
- `Sistemas de archivos (File Systems)`
- `Servicios misceláneos (Miscellaneous Services)`
- `Modo de inicio (Boot Mode)`

La sección *trabajo en red* se divide además en tres subcategorías: *Tareas del cliente* (`Client Tasks`), *Tareas del servidor* (`Server Tasks`) y `Misc`. Esta sección aborda muchas tareas del trabajo en red, como configuración del cliente DNS y del servidor, exportación del sistema de archivos NFS, configuración del servidor Web Apache, y otras. La tabla 10.1 lista todas las tareas disponibles bajo esta categoría de *trabajo en red*.

Tabla 10.1 Funciones disponibles de la categoría del trabajo en red.

Categoría principal	Subcategoría	Descripción
Tareas del cliente		
	Información básica del host	Modifica el nombre del *host* del sistema local y suministra acceso a los parámetros de configuración de la interfaz
	Especificación del nombre del servidor	Modifica el archivo de configuración `/etc/resolv.conf` de resolución DNS
	Enrutado y puertas de enlace	
	Ruta de búsqueda del nombre del host	Dice al sistema el orden en el cual se sondearán los diferentes servicios de nombre; modifica el archivo de configuración `/etc/host.conf`
	Servicios de información de la red (NIS)	Configura el cliente NIS
	Configuración de interfaz IPX	Habilita el protocolo de red IPX en interfaces específicas de red
	PPP/SLIP/PLIP	Configura los servicios de capa de red en puertos paralelos o en serie; usa el panel de control de red estándar

Tabla 10.1 Funciones disponibles de la categoría de trabajo en red. (*continuación*)

Categoría principal	Subcategoría	Descripción
Tareas del servidor		
	Sistema de archivos exportado(NFS)	Configura los sistemas de archivos disponibles para los *hosts* clientes
	Alias IP para hosts virtuales	Crea pseudo-interfaces
	Servidor de nombres de dominio (DNS)	
	Config	Contiene varios subelementos para configurar un servidor DNS
	Adicionar/Editar	Contiene subelementos para adicionar nuevos *hosts* o modificar los *hosts* existentes en la base de datos DNS
	Seguridad	Contiene subelementos para controlar cuáles *hosts* pueden obtener información de zona DNS
	Servidor Web Apache	
	Predeterminados	Controla cualquier opción de configuración predeterminada, como dirección de correo electrónico del administrador, nombres de los archivos de registro de error, y cuál puerto TCP escuchar; incluye muchas otras opciones

Tabla 10.1 Funciones disponibles de la categoría de trabajo en red. (*continuación*)

Categoría principal	Subcategoría	Descripción
	Dominios virtuales	Suministra los medios para crear dominios virtuales de red con servidores múltiples Web
	Subdirectorio specs	Permite la configuración de rutas URL y directorios en el servidor Web
	Servidor de archivos Samba	
	Predeterminados	Controla muchas opciones de configuración predeterminada para el servidor Samba
	Configuración predeterminada para el directorio nogar del usuario	Configura el compartir la partición de un disco o del sistema de archivos
	Configuración predeterminada para impresoras	Subelemento no funcional; dar un clic a este elemento no hace nada
	Disco compartido	Crea puntos de entrada múltiples e independientes para un servidor de archivos Samba
	Sistema de entrega de correo (sendmail)	
	Básico	Configura la operación básica del servidor sendmail
	Filtros anti-spam (publicidad no deseada)	Controla la creación de filtros de correo que incluyen servicios de transmisión

Tabla 10.1 Funciones disponibles de la categoría de trabajo en red. (*continuación*)

Categoría principal	Subcategoría	Descripción
	Servidor Ftp (wu-ftpd)	
	Configuración básica	Configura la dirección de correo electrónico del administrador, archivo *banner*, archivo de mensajes de apagado y otras opciones de control
	Hosts virtuales	Controla la creación de *hosts* virtuales usando el servidor FTP
Misc	Información acerca de otros hosts	Suministra acceso al archivo /etc/hosts
	Información acerca de otras redes	Suministra acceso al archivo /etc/networks
	Acceso de red Linuxconf	Crea filtros de *host* para restringir el acceso al proceso del servidor de configuración Linux

La categoría cuentas de usuario (User Accounts) tiene cuatro subcategorías: Normal, cuentas especiales, alias de correo electrónico y políticas. Esta categoría suministra muchas funciones para la administración de cuentas UNIX y tareas asociadas, incluidas la capacidad de crear nuevas cuentas del sistema o modificar las existentes, mantener los alias de correo y controlar las contraseñas. La tabla 10.2 suministra una lista completa de estas funciones.

Tabla 10.2 Funciones disponibles de la categoría cuenta de usuario.

Categoría principal	Subcategoría	Descripción
Normal		
	Cuentas del usuario	Adiciona, borra o remueve cuentas individuales
	Definiciones de grupo	Adiciona, borra o remueve grupos individuales
	Cambio de la contraseña de root	Cambia la contraseña de root
Cuentas especiales		
	Cuentas PPP	Crea/modifica cuentas especiales del sistema para el acceso PPP
	Cuentas SLIP	Crea/modifica cuentas especiales del sistema para el acceso SLIP
	Cuentas UUCP	Crea/modifica cuentas especiales del sistema para el acceso UUCP
	Cuentas POP (sólo correo)	Crea/modifica cuentas especiales del sistema para el acceso de correo POP
	Cuentas virtuales POP (sólo correo)	Crea/modifica cuentas especiales del sistema para el acceso de correo virtual POP

Tabla 10.2 Funciones disponibles de la categoría cuenta de usuario. (*continuación*)

Categoría principal	Subcategoría	Descripción
Alias de correo electrónico		
	`Alias de usuario`	Mantiene los alias de correo del sistema, los cuales se almacenan en el archivo `/etc/aliases`
	`Alias de dominio virtual del usuario`	Mantiene los alias de correo del sistema virtual
Políticas		
	`Políticas de contraseña y cuenta`	Controla varios parámetros respecto a las contraseñas del sistema de usuarios regulares y de las cuentas
	`Shells disponibles del usuario`	Controla cuál intérprete de comandos es el shell de inicio de sesión predeterminado
	`Shells disponibles PPP`	Controla cuál intérprete de comandos es para un usuario PPP
	`Shells disponibles SLIP`	Controla cuál intérprete de comandos es para un usuario SLIP
	`Mensaje del día`	Configura una cadena mensaje del día para que sea presentada durante una sesión de inicio normal

La sección *sistemas de archivos* (`File systems`) contiene elementos adicionales que ayudan a controlar y mantener sistemas de archivos. Los `servicios misceláneos` controlan los niveles de ejecución predeterminados Linux mientras que el elemento de `modo de inicio` se usa para la configuración de arranque de nivel inferior Linux y otros parámetros. Todas estas tres categorías se incluyen en la tabla 10.3.

La categoría raíz `Control` también contiene subcategorías que se usan para tener acceso a parte de las funciones del panel de control descritas aquí y un número de tareas de administración, como montar los sistemas de archivos y la configuración de los archivos del sistema:

- `panel de control`
- `control de archivos y sistemas`
- `registros`

Tabla 10.3 Funciones adicionales de la categoría **config.**

Categoría principal	Subcategoría	Descripción
Sistemas de archivos	`Acceso a la unidad local`	Hace cambios al archivo `/etc/fstab` para sistemas de archivos locales
	`Acceso al volumen nfs`	Hace cambios al archivo `/etc/fstab` para volúmenes NFS
	`Configura archivos y particiones de intercambio`	Hace cambios al archivo `/etc/fstab` para particiones de intercambio
	`Establecer cuota predeterminada`	Controla los parámetros para cuotas de disco
	`Verificar algunos permisos de archivo`	Verigica permisos de archivos críticos en el directorio `/etc`
Servicios misceláneos		

Tabla 10.3 Funciones adicionales de la categoría **config.** (*continuación*)

Categoría principal	Subcategoría	Descripción
	Servicios iniciales del sistema	Controla el establecimiento del nivel de jecución predeterminado
Modo de arranque		
	Lilo	
	Predeterminaciones LILO (cargador de arranque Linux)	Controla los parámetros relacionados con el arranque del sistema Linux
	Configuraciones LILO Linux	Controla los parámetros relacionados con las opciones del *kernel*, cuando se arranca
	Configuraciones de LILO con otros OS	Define otras etiquetas de sistemas operativos y partición de arranque
	Configuración de arranque Predeterminada	Selecciona los parámetros de arranque que se deben usar cuando se arranca el *kernel*
	Jn kernel nuevo	Configura el arranque de un *kernel* Linux existente
	Un kernel que un usuario cumplió	Configura el arranque de un nuevo *kernel* Linux
	Modo	
	Modo predeterminado de arranque	Controla cómo debería arrancar el sistema: modo de texto y red frente a gráfico y red. Suministra parámetros de control adicionales

La tabla 10.4 lista todos los elementos de las subcategorías bajo estas categorías principales.

Tabla 10.4 Funciones disponibles bajo la categoría `Control`.

Categoría principal	Subcategoría	Descripción
Control		
	`Panel de control`	
	`Activar la configuración`	Habilita o inhabilita los cambios hechos al sistema; además, puede presentar los cambios antes de llevarlos a cabo
	`Apagar/reiniciar`	Corre el comando **/usr/bin/shutdown**
	`Control de los servicios`	Habilita/inhabilita los servicios del sistema, es decir, aquellos iniciados por el sistema durante el arranque normal
	`Montar/desmontar los sistemas de archivos`	Lo mismo que el elemento de control de servicios anterior
	`Control configurado de las unidades locales`	Lo mismo que el elemento de control de servicios ya citado
	`Control configurado en los volúmenes nfs`	Lo mismo que el elemento de control de servicios ya citado
	`Montar otros sistemas de archivos NFS`	Lo mismo que el elemento de control de servicios ya citado
	`Configura tareas planeadas de superusuario`	Mantiene una entrada CRONTAB para la cuenta root

Tabla 10.4 Funciones disponibles bajo la categoría `Control`. (*continuación*)

Categoría principal	Subcategoría	Descripción
	`Archivar configuraciones`	No implementada
	`Conmutador del perfil del sistema`	Controla el uso de uno o más perfiles de sistema que se pueden activar en el comando
	`Controlar los enlaces PPP/SLIP/PLIP`	Habilita o inhabilita los enlaces de red
	`Controlar los archivos y sistemas`	
	`Configurar todos los archivos de configuración`	Suministra la capacidad para mantener archivos importantes de configuración del sistema. Los archivos específicos incluyen `/etc/HOSTNAME`: `/etc/aliases`. y `/etc/exports`.
	`Configurar todos los comandos y demonios`	Configura la tranformación entre los nombres de comandos del sistema y los comandos. Por ejemplo, **cat** se transforma en `/bin/cat`
	`Configurar permisos y la propiedad del archivo`	Controla el permiso de los archivos y de los directorios a través del sistema
	`Configurar módulos Linuxconf`	Administra los módulos Linux. Cada uno se puede activar o desactivar

Tabla 10.4 Funciones disponibles bajo la categoría `Control`. (*continuación*)

Categoría principal	Subcategoría	Descripción
	`Configurar los perfiles del sistema`	Alterna entre uno y más perfiles de configuración del sistema
	`Sobreponer adiciones Linuxconf`	Controla los módulos Linux
	`Crear adiciones _inuxconf`	Define un nuevo módulo Linux
	`registros`	
	`Mensajes de inicio`	Presenta información de arranque a través del archivo de registro del arranque
	`Registros Linuxconf`	Presenta el archivo de registro Linuxconf
	`Fecha y tiempo`	Controla las opciones de la fecha y hora del sistema
	`Características`	Controla el comportamiento especial del programa Linuxconf

11

Herramienta tkined - Editor de la red

El editor de la red **tkined** disponible con el paquete de red Scotty suministra una interfaz gráfica de usuario (GUI) para un gran número de funciones de seguimiento y soporte de red. También proporciona un editor interactivo para crear y manipular mapas de topología de la red. La herramienta es personalizable por el usuario y se pueden integrar en el paquete básico herramientas adicionales de trabajo en red y del sistema. El software se basa en el sistema XWindow y está disponible y es soportado en varias plataformas diferentes de sistema operativo.

Vista general de tkined

La aplicación **tkined** es un administrador de la red completamente funcional que suministra muchas de las capacidades de administración que se encuentran en algunos de los productos comerciales más populares. Esta herramienta es considerada uno de los mejores paquetes de seguimiento de red de dominio público disponible y ostenta muchas características y funciones. La aplicación **tkined** no es un paquete de software complicado, pero contiene muchos elementos de menú, submenús, subventanas y módulos específicos. Por tanto, se deben revisar los fundamentos para empezar a usar el software.

Desde el punto de vista funcional, **tkined** hace lo siguiente:

- descubre dispositivos y sistemas de trabajo en red
- crea/mantiene mapas y vistas de red
- administra dispositivos SNMP
- soporta el seguimiento IP
- crea gráficos y cuadros de rendimiento

Para arrancar la aplicación, se invoca el comando **tkined** en un indicador de comandos shell:

```
# /usr/bin/tkined
```

Al ejecutar el comando se presentará la ventana principal, que se muestra en la figura 11.1. De manera predeterminada, la ventana no contiene ningún objeto o dispositivo de red, porque se debe llevar a cabo una cierta cantidad de personalización antes de que **tkined** pueda hacer seguimiento a su red.

La ventana principal contiene una dotación completa de elementos de menú, una paleta de selección de herramientas a la izquierda y un panel de transformación de topología de red. El panel de transformación de red se usa para presentar una variedad de elementos de red en formato de icono. El paquete contiene una gran cantidad de iconos que se pueden usar para representar dispositivos y sistemas específicos, o topologías específicas de red, como bus y anillo, que se puedan encontrar en una red. Finalmente, se pueden crear diferentes vistas que contienen dispositivos específicos, subredes completas, cuadros y gráficos, y otros elementos.

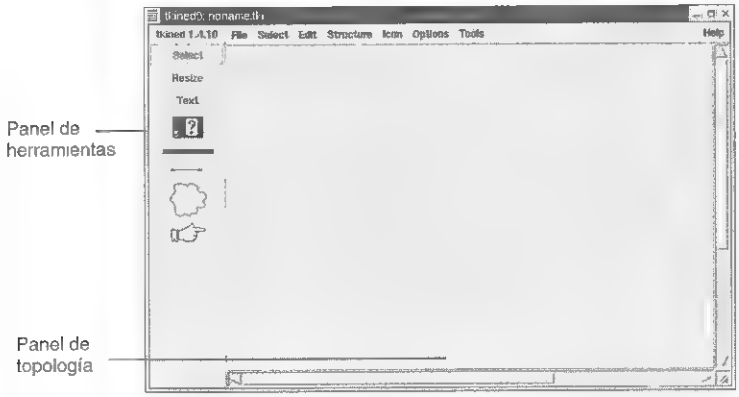


Figura 11.1 Ventana principal **tkined**.

Vistas de topología

El paquete da soporte a la idea de usar diferentes vistas para representar varios aspectos de la red, como enrutadores Cisco o varios servidores críticos. Se pueden crear vistas adicionales para alojar gráficos de rendimiento o subredes completas. Cuando la aplicación es arrancada por primera vez, se crea una vista vacía. El nombre de la vista se presenta en el encabezado de la ventana principal de la aplicación. El nombre de la vista es también el nombre del archivo en el cual se almacena. El archivo de vista predeterminado es `noname.tk`, el cual se puede cambiar.

El propósito principal de las vistas es ayudar a dividir la red en componentes más pequeños. La cantidad de sistemas en muchas redes hace muy difícil hacerles un seguimiento a éstas, pero, particinar la red en elementos o compartimentos diferentes, puede facilitar la tarea. La opción de cómo dividir la red es realmente del usuario. Se debe recordar que ésta se debería particionar en áreas lógicas o funcionales que se ajusten mejor al modelo del departamento o empresa.

Por ejemplo, supóngase que el usuario trabaja para el grupo de red cuya principal responsabilidad es soportar la red vertebral y la red de área amplia (WAN). El soporte para subredes específicas se apoya en los departamentos individuales, lo cual significa básicamente que los dispositivos pueden ser adicionados, movidos o removidos por estas subredes en cualquier momento. En consecuencia, puede no ser razonable hacer seguimiento a nodos finales específicos o a dispositivos contenidos en esas subredes;

en cambio, sería más interesante hacer seguimiento a los dispositivos centrales vertebrales y a los componentes WAN. Los mapas o vistas de topología creados en **tkined** reflejarían esta fisiología de seguimiento.

De hecho, el mayor interés de todos los participantes de cada grupo individual o de soporte es instalar copias adicionales de **tkined** para abordar cada necesidad específica y requerimientos de seguimiento. Usar copias múltiples de **tkined** en el mismo sistema es una de las aproximaciones posibles a esta situación.

Descubrimiento de la red

Un uso valioso de **tkined** es poder descubrir dinámicamente los dispositivos de red en una sola subred, subredes múltiples o dispositivos individuales. Uno de los objetivos principales para un administrador de red es integrar en la herramienta, rápida y fácilmente, la topología de la red. Para comenzar el proceso de descubrimiento, se selecciona el submenú `IP Discovery` (descubrir IP) del menú principal `Tools` (herramientas). Un nuevo menú llamado `IP Discovery` se presentará a la derecha del menú `tool`. La figura 11.2 muestra este nuevo menú, el cual contiene una opción, `Discover IP Network` (descubrir red IP), que consulta los dispositivos usando los protocolos ICMP y SNMP, en un intento por aprender la topología de la red. Al seleccionar esta opción se presenta la subventana que se muestra en la figura 11.2.

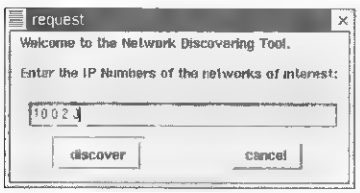


Figura 11.2 Ventana de solicitud de descubrimiento de red.

La ventana de diálogo requiere que el usuario introduzca una dirección de red IP válida o un grupo de direcciones seguidas de espacios. Como se muestra en la figura 11.2, la red `10.0.2.0` se descubrirá cuando se dé clic al botón discover. Dar clic también presenta la ventana que se muestra en la figura 11.3.

La ventana muestra el progreso del descubrimiento y los pasos específicos involucrados. El proceso de descubrimiento que se muestra en la figura 11.3 sólo tomó 66 segundos para completarse, y encontró siete nodos en la red. Dado el número de

nodos, el tiempo requerido no representa un problema. Sin embargo, en una red grande, el proceso de descubrimiento podría tomar mucho más tiempo. Nótese que **tkined** ha detectado aquellos dispositivos que soportan SNMP y aquellos que no lo hacen. En el ejemplo, la aplicación encontró seis agentes SNMP, lo cual es correcto para esta red. Se verá más adelante que **tkined** soporta un número significativo de funciones basadas en SNMP, y que es primordial una detección certera de los dispositivos SNMP. La otra información importante respecto a la ventana de estado es la cuenta del número de objetos creados. La figura 11.3 indica que se crearon en total 16 objetos. Puesto que hubo siete dispositivos, cada uno tiene un icono individual, un objeto de conexión de red y dos redes.

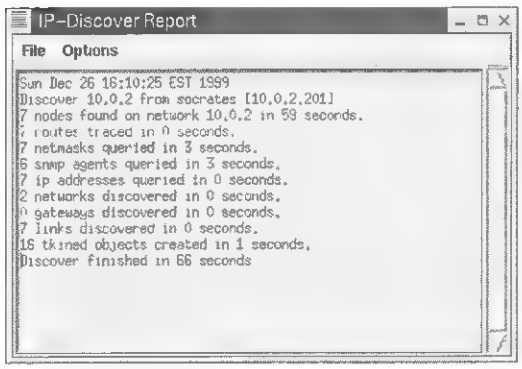


Figura 11.3 Ventana de estado de descubrimiento.

Una vez se completa el descubrimiento, **tkined** intenta representar la topología y los dispositivos con el panel de topología de trabajo en red. La figura 11.4 muestra los resultados de sondear la red `10.0.2.0`. Nótese que los iconos están un poco desorganizados y que el icono predeterminado se usa para todos los dispositivos. Se puede personalizar el mapa de topología y mover los objetos alrededor para hacer el mapa un poco más atractivo visualmente. Infortunadamente, si quiere usar iconos más representativos, debe asignarlos manualmente.

Para mover un icono alrededor, simplemente se selecciona el objeto y luego, presionando ambos botones del ratón, se mueve el icono a la posición deseada y se libera el ratón. Para ayudar a rastrear las acciones del usuario, el software presenta un cuadro como una guía que se mueve a medida que se desplaza el ratón. El usuario puede mover de esta manera todos los objetos de un lado para otro en la vista de topología.

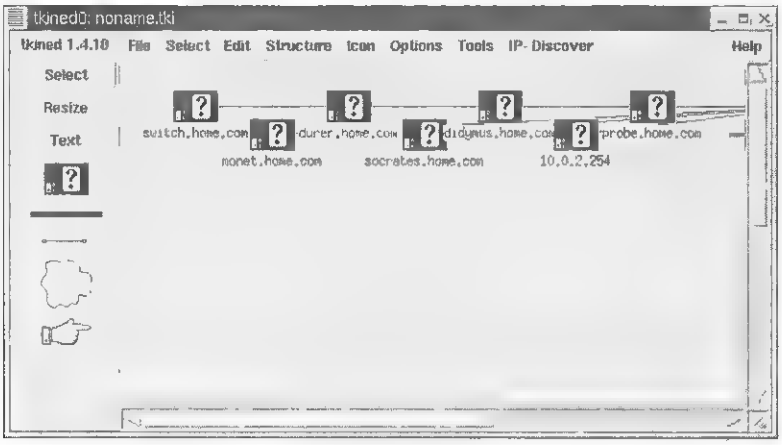


Figura 11.4 Topología descubierta de la red.

Para remplazar el icono predeterminado para un objeto, se selecciona el objeto y se va al menú `Icon`, luego se da un clic en la opción `Node`. Cuando aparece una lista de tipos de sistema, se selecciona aquella que coincide con el dispositivo que se está cambiando, como se muestra en la figura 11.5. Después de esto, el icono del dispositivo cambiará al tipo especificado. Como se ve en la lista, una gran cantidad de iconos diferentes está disponible para los sistemas. El menú también contiene opciones adicionales que tienen iconos para los dispositivos de trabajo en red, como los conmutadores y los enrutadores. La opción `Network device` contiene iconos para un puente, un concentrador, un conmutador, un enrutador, un servidor de terminales y un enrutador Cisco. La opción `Peripherals` tiene iconos para los siguientes dispositivos: módem, teléfono, impresora, impresora láser, trazador HP, HP Deskjet y HP Scanjet. La opción `Misc` contiene iconos genéricos, como cuadro pequeño, cuasro normal, cuadro grande y conector.

La herramienta es también suficientemente flexible para permitir usar iconos personalizados y aplicarlos a objetos. Consúltese la sección "Personalización" para obtener información adicional. Despues de remplazar todos los iconos predeterminados con unos más representativos, y de mover los objetos alrededor para hacer el mapa más legible, éste debería representar más exactamente la red. El descubrimiento de la red del ejemplo previo produjo la vista de topología que se muestra en la figura 11.6.

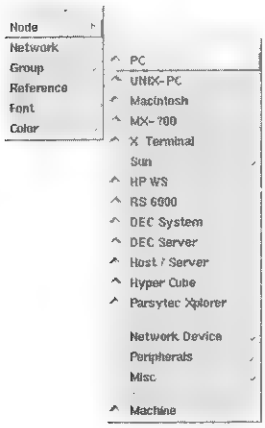


Figura 11.5 Lista de iconos de dispositivo.

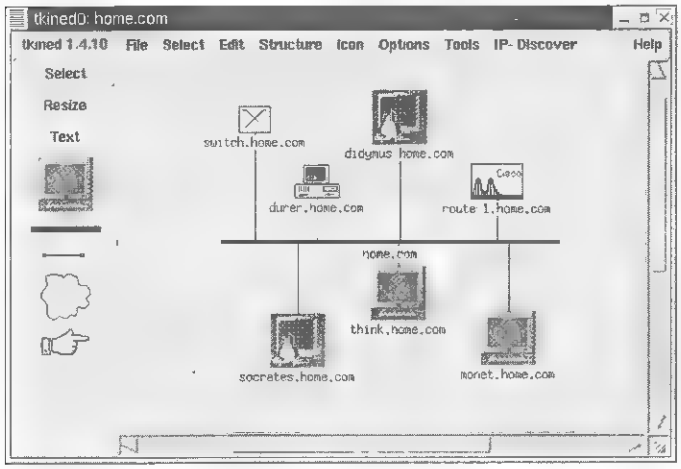


Figura 11.6 Vista de topología completada.

En la vista de topología home.com de esta red de prueba, cada uno de los iconos de los dispositivos se han cambiado para que corresponda al dispositivo actual que representa. Nótese que los sistemas Linux usan un icono personalizado que se adicionó a **tkined** después de la instalación. Este icono no se encontrará en la lista estándar de iconos disponibles. Téngase en cuenta que se puede asignar cualquier icono de sistema a un objeto de dispositivo, sin importar el objeto; ningún mecanismo interno o regla impedirá esta acción. La única excepción a esto se presenta cuando busca asignar iconos diferentes a tipos incompatibles. Por

ejemplo, la herramienta no permitirá que un tipo de red (como `Ethernet 10MB`) se asigne a un icono de dispositivo, y viceversa. El tipo de red `Ethernet 10MB` representa una línea horizontal que conecta uno o más nodos.

En ciertas situaciones, esto puede plantear un problema, porque si un icono para un dispositivo no representa el dispositivo, puede ser tomado por otro. En la práctica, este problema se puede atenuar, debido a que las acciones para un dispositivo deberían basarse no sólo en la imagen del icono, sino también en la etiqueta del dispositivo.

Aquí surge la pregunta: "¿Por qué se debería usar el autodescubrimiento si se tiene que asignar los iconos y también reordenar la vista? Bueno, ésta es una buena pregunta, Sin embargo, hay más en el proceso de descubrimiento de lo que ven nuestros ojos; la herramienta hace un buen trabajo de construcción de una lista de todos los dispositivos de la red, y crea objetos individuales para ellos. También asigna a cada uno una etiqueta que es el nombre de *host* del dispositivo. Además, detecta cuáles dispositivos soportan también SNMP y asigna parámetros predefinidos para la racolección de información SNMP.

No obstante, se puede preferir configurar manualmente las vistas de topología de la red. La siguiente sesión mostrará cómo hacerlo.

Uso del panel de herramientas

El editor **tkined** suministra varias herramientas en el lado izquierdo de la ventana principal, que se pueden usar para construir mapas de topología o hacer cambios a los mapas de topología existentes. La figura 11.7 muestra una vista recortada de la paleta de herramientas de la ventana principal.

La paleta de herramientas contiene las siguientes herramientas:

- Selección
- Cambio de tamaño
- Texto
- Dispositivo
- Red
- Conexión
- Grupo
- Referencia

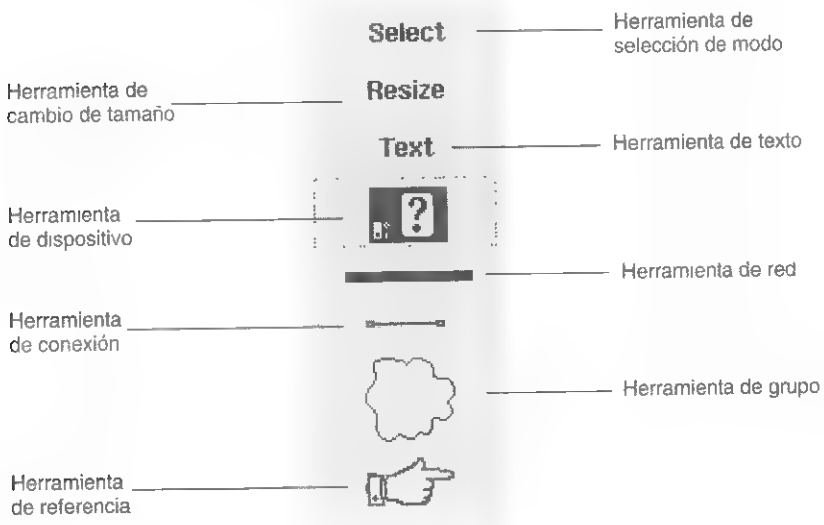


Figura 11.7 Vista del panel de herramientas.

Herramienta de selección

Este elemento es realmente más un modo que una herramienta; habilita un modo de selección que indica si el ratón se usará para seleccionar objetos existentes en la vista de topología. Se necesitará usar este modo cuando se cambien los atributos del objeto y los parámetros o se borren los objetos. Habilitar este modo cambia el puntero del ratón a una flecha negra apuntando arriba y a la izquierda ( ). Se puede seleccionar un objeto individual dándole un clic izquierdo. Para seleccionar un grupo de objetos, se da un clic izquierdo al ratón y se arrastra sobre los objetos que se desea seleccionar. En este punto, el puntero del ratón cambiará a una ( ), y se mostrará un rectángulo o una banda elástica, el cual incluirá todos los objetos seleccionados.

Para borrar un objeto, se selecciona éste, y luego se selecciona la opción `Delete` del menú `Edit` o se usa el comando equivalente de teclado, **^d** (**control+d**).

Herramienta de cambio de tamaño

Esta herramienta suministra la capacidad de dar otro tamaño a la topología de red y a los objetos gráficos. Para agrandar o reducir uno de estos objetos, primero se da un clic izquierdo al objeto. A continuación, se selecciona la herramienta de cambio de tamaño y se da un clic en una de las cuatro manijas de objeto,

se arrastra el objeto a la posición deseada y se suelta el botón del ratón.

Herramienta de texto

Esta herramienta agrega objetos de texto a la vista de topología. Adicionar información estática de texto puede mejorar la apariencia de la vista y suministra información vital. Por ejemplo, si la vista incluye diferentes subvistas que contienen múltiples subredes, se pueden agregar etiquetas específicas a cada una de las subredes. Cuando se selecciona esta herramienta, el puntero del ratón cambia a punto de inserción (I). Estos objetos se pueden controlar como se haría con cualquier otro elemento asociado en una vista particular.

Para agregar un objeto textual, se resalta el cuadro de texto, seleccionándolo. A continuación, se da un solo clic al botón izquierdo en el ratón, en el área de la vista de topología y se comienza a digitar la etiqueta. Si se quiere editar una vista existente, se da clic al botón derecho del ratón en el punto del texto donde se quiere hacer la modificación.

Herramienta de dispositivo

Cuando se selecciona, esta herramienta permite al usuario crear objetos de dispositivos. Ésta es la forma principal para añadir dispositivos adicionales a un mapa de topología. Si se selecciona esta herramienta, el puntero del ratón cambia a un signo más (+). El icono es el gráfico predeterminado del dispositivo, el cual se puede cambiar seleccionando otro icono del menú `Icon`.

Para crear un objeto, simplemente se da clic izquierdo sobre la vista de topología, una vez el puntero del ratón ha cambiado al signo más (+).

Herramienta de red

Esta herramienta crea iconos de red que interconectan objetos de dispositivo. El icono estándar de topología de red es el bus, el cual no se puede cambiar. Sin embargo, el espesor que representa una topología de red en bus se puede cambiar, dependiendo del tipo de red especificado. La opción `Network` en el menú `Icon` se utiliza para cambiar los iconos de topología de red localizados en la vista de topología.

Para crear un tipo de objeto de red, se selecciona el objeto de red con el botón izquierdo del ratón. A continuación, se mueve el ratón sobre el área de la vista de topología. Luego, se da un

clic izquierdo para marcar el comienzo de la red, después se arrastra a la derecha hasta que la longitud deseada de la red (o línea) se haya establecido.

Herramienta de conexión

La herramienta de conexión hace conexiones entre objetos de dispositivos individuales y uno o más objetos de topología de red. Los objetos de los dispositivos se pueden interconectar a uno o más objetos de red, pero el uso de la herramienta de conexión para enlazar dos objetos de red no está soportado. La única forma de interconectar los elementos de la red se logra usando uno o más objetos de dispositivo.

Para conectar un objeto de dispositivo y un elemento de red, se selecciona la herramienta de conexión, luego hay que dirigirse al área de topología. Nótese que el puntero del ratón cambiará a (+). A continuación, se da un clic izquierdo en el objeto del dispositivo y se comienza a arrastrar el puntero hacia el objeto de red que se quiere conectar. El software presentará una línea de conexión que es rastreada hacia el ratón. Luego se da un clic izquierdo para completar la operación.

Herramienta de grupo

Esta herramienta se usa principalmente para combinar un grupo de objetos individuales en un único objeto. Por ejemplo, una colección de estaciones de trabajo se puede organizar en vistas de subred. Por tanto, la vista de nivel superior de topología puede incluir un grupo de objetos que contiene objetos de dispositivo adicionales u otros elementos.

Para crear un nuevo grupo, simplemente se escogen los elementos que se deberían incluir en el grupo, más el nievo grupo de objetos, y se selecciona `Join group` en el menú `Structure`. Para tener acceso a los miembros del grupo de objetos se debe expandir el grupo. La figura 11.8 muestra dos grupos llamados `Subnet 134.110.1.0` y `Subnet 134.110.2.0`. En este ejemplo, el icono predeterminado para cada grupo se ha cambiado para que represente una topología de red bus.

Para expandir un grupo específico, se selecciona el grupo de elementos y se ejecutá la opción `expand` en el menú `Structure`. Los iconos que estuvieron ocultos una vez ahora se muestran en la vista, la cual se presenta en la figura 11.9. La facilidad de agrupar puede organizar una red grande. El icono asociado con un grupo también se puede cambiar de la misma manera que los

objetos de dispositivos individuales. Los iconos de grupo disponibles se listan a través de la opción `Group`, en el menú principal `Icon`.

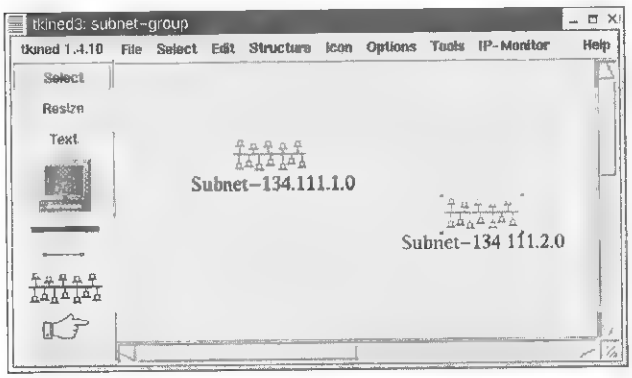


Figura 11.8 Ejemplo de vista que contiene grupos de objetos.

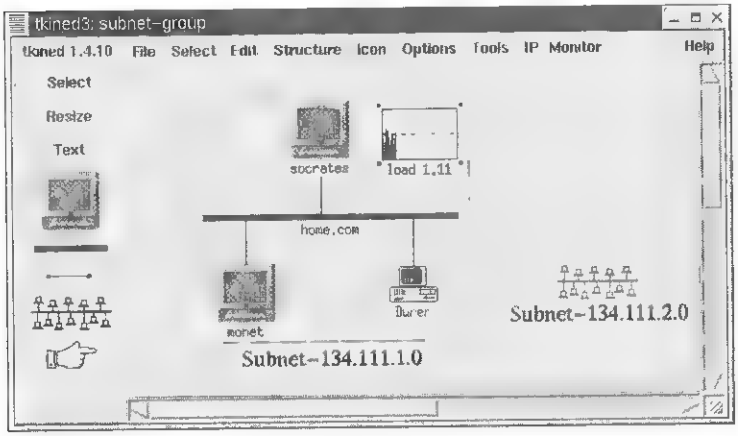


Figura 11.9 Grupo expandido.

Herramienta de referencia

Esta herramienta crea enlaces con vistas existentes de topología a las que se puede tener acceso usando un elemento individual de icono. Los objetos referenciados suministran un atajo hacia otra instancia de una vista **tkined**. La vista de topología que se presenta en la figura 11.10 muestra tres enlaces de referencia diferentes: `Accounting Subnet`, `Performance` y `Backbone`.

Para tener acceso a la vista que está detrás del enlace de referencia, se da un clic derecho al icono y se selecciona la opción

`Open (this view)`. Esto presentará la vista que está enlazada a este objeto de referencia sobre el panel existente. La figura 11.11 muestra un ejemplo de la vista asociada con el objeto de referencia `Accounting Subnet`.

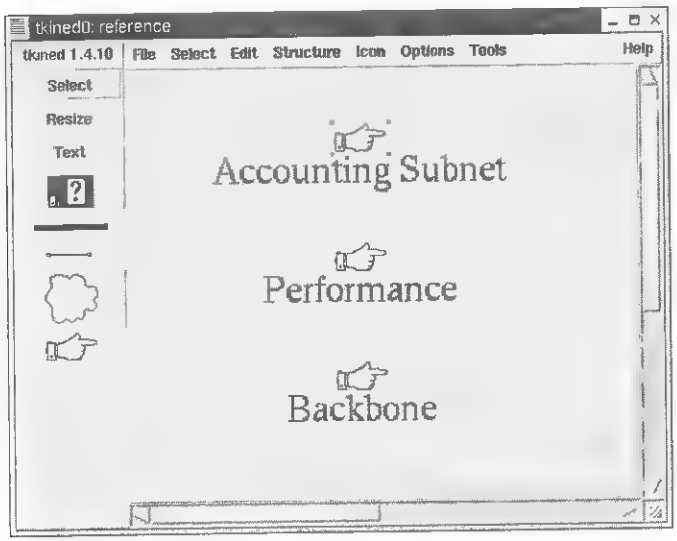


Figura 11.10 Objetos de referencia.

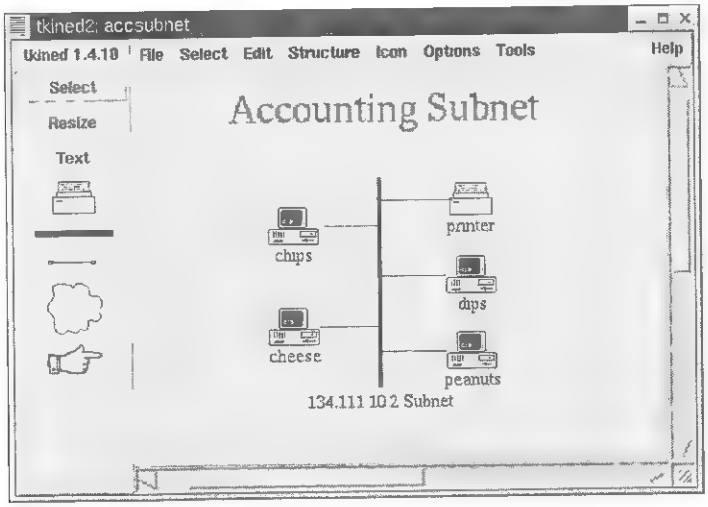


Figura 11.11 Vista referenciada.

Construcción de vistas

Cuando se construyen vistas de topología, se debe usar la paleta de herramientas analiza en la sección anterior. Esta sección suministra instrucciones paso a paso para crear vistas personalizadas de red que coincidan con las topologías física o lógica de la red del usuario. Están disponibles dos aproximaciones mayores de cómo diseñar una vista de topología: *física* y *lógica*. Una vista de topología física imita la forma en que una red física está instalada o conectada.

Por ejemplo, considérese una red Ethernet que contiene un único conmutador LAN con varias estaciones de trabajo interconectadas a éste. Se puede hacer un mapa de todos los elementos físicos de esta red, incluido el cableado que conecta cada uno de los nodos al conmutador, como se muestra en la figura 11.12. A partir del diagrama de la red, se pueden determinar las tomas de oficina usadas por cada estación de trabajo, la ubicación del cable que cada toma de oficina usa y la transformación entre un puerto del conmutador y una estación de trabajo individual. Los números encerrados en los rectángulos más cerca de los dispositivos representan las tomas de la oficina, mientras que los otros cuadros son localizaciones del panel de conexión. Normalmente, se presentan muchos detalles en un diseño físico de red, como la localización del equipo (el conmutador, por ejemplo) y otra información asociada. De manera usual, los diagramas físicos de red son los más difíciles de mantener, porque cualquier pequeño cambio, como el que se agregue o borre una estación de trabajo, puede hacerlos inexactos.

El administrador de la red debe ser diligente en mantener esta información actualizada, lo cual puede ser una ocupación de tiempo completo.

Una vista lógica de red por lo general presenta a la red en un nivel mucho más amplio y alto. Las vistas lógicas muestran la relación de los nodos y los dispositivos de la red desde un nivel puramente conceptual, y no necesariamente incluye detalles respecto a las conexiones físicas, a la transformación puerto conmutado, y otra información. La figura 11.13 muestra un ejemplo de este tipo de topología.

Es usual que muchos administradores de red usen los dos enfoques cuando construyen vistas. Por tanto, las vistas de topología de red empiezan en un nivel superior y, a través de una serie de vistas, llegan a ser cada vez más detalladas.

Antes de construir una topología, también se recomienda determinar cuál tipo de topología, lógica, física o ambas, se usará.

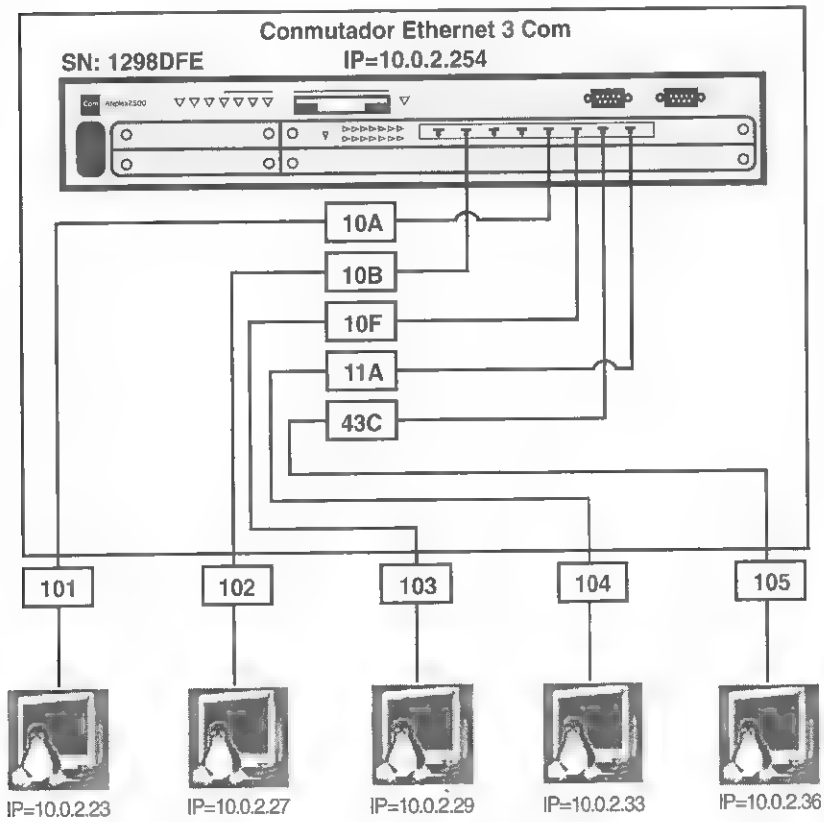


Figura 11.12 Ejemplo de la topología física.

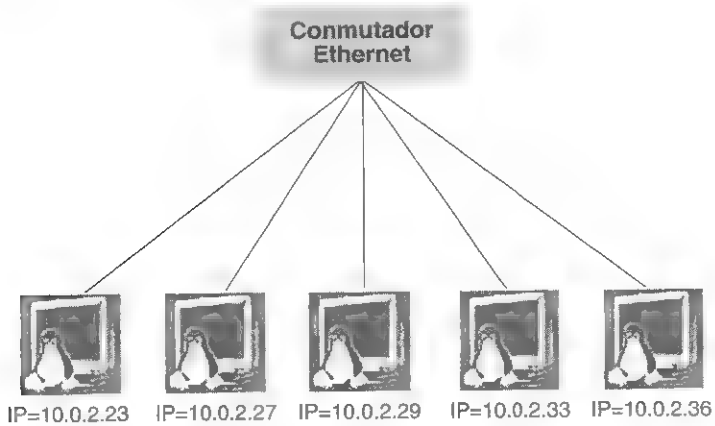


Figura 11.13 Ejemplo de la topología lógica.

Además, en un esfuerzo para asegurar que la vista de topología sea efectiva y útil, es importante considerar las formas en las cuales las redes son vistas y soportadas en el entorno local. Por ejemplo, una gran corporación puede delegar autoridad administrativa para cada una de las subredes a cada uno de los departamentos. Como resultado, el grupo central de red se enfoca principalmente en la red vertebral corporativa. Por tanto, se construirá una vista de topología sólo para esta porción de la red, mientras que cada uno de los propietarios individuales de las subredes mantiene una vista de topología sólo para su porción de la red.

Usando **tkined**, se construirá un conjunto de vistas lógicas que representa uno de los enfoques más comunes para describir la red de una corporación de tamaño mediano. En este ejemplo, la corporación emplea un grupo de servicios de administración de información (MIS) responsable de todos los aspectos del trabajo en red en la compañía. A cada departamento se le asigna su propia subred, pero el grupo de red es responsable de mantener la conectividad entre los nodos en las subredes y entre cada subred y la red vertebral. La compañía tiene su base en Boston, Massachusetts, y está situada en un campus de tres edificios, todos muy cercanos uno del otro. El grupo también mantiene una WAN para conectar muchas oficinas de soporte y ventas.

Puesto que la red incluye ambos componentes, LAN y WAN, es necesario construir varias vistas. Nótese que el paso incluido en esta sección involucra la creación de objetos para dispositivos reales en la red. Se deberían sustituir los dispositivos específicos por los dispositivos reales disponibles en la red. La construcción de cada vista requerirá muchos pasos individuales. Las secciones que siguen suministran instrucciones paso a paso para construir una vista LAN y una vista WAN.

Vista LAN

Para construir una vista LAN se siguen estos pasos:

- Se crea un nuevo directorio que contenga todas las vistas. En este ejemplo, el directorio `/home/network-views` se crea con este propósito.
- Se obtiene una vista limpia (nueva) seleccionando la opción `New View` en el menú `File`. Esto presentará una vista vacía. Si se no se quiere crear una vista nueva, solamente se borran los objetos indeseados, con el fin de crear espacio para los objetos adicionales.

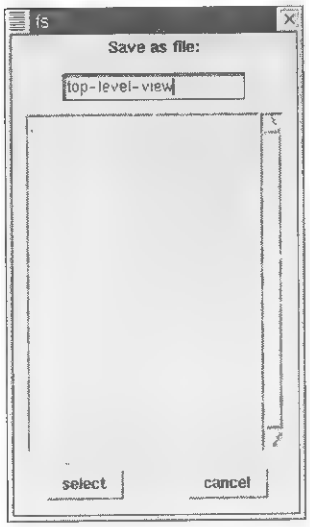


Figura 11.14 Ventana de diálogo Guardar como.

- Puesto que se usa la vista predeterminada de `noname.tki`, se cambia el nombre de la vista a `top level-view` seleccionando la opción `Save as...` en el menú `File`. Aparecerá una ventana de diálogo, como se muestra en la figura 11.14. A continuación se ingresa el nombre de la vista nueva y se da clic en el botón `Select`.
- Usando el botón izquierdo del ratón, se selecciona la herramienta Group (grupo). Un borde transparente deberá aparecer alrededor de esta herramienta, indicando que está activa. Cuando el ratón se mueve en la región de la vista de topología, el puntero del ratón cambia a un símbolo.
- Se da un solo clic al botón izquierdo del ratón para seleccionar el primer objeto del grupo. A continuación, se da un clic derecho en el mismo objeto y se selecciona la opción `Edit all Attributes...` del menú contextual (*pop-up*). Aparecerá una nueva ventana de diálogo, como se muestra en la figura 11.15. Se puede cambiar la etiqueta (o el nombre del objeto) en esta ventana. De manera predeterminada, contiene la etiqueta de `group1`. A continuación, se introduce la nueva etiqueta de `LAN` y se da clic al botón `Set Values`.

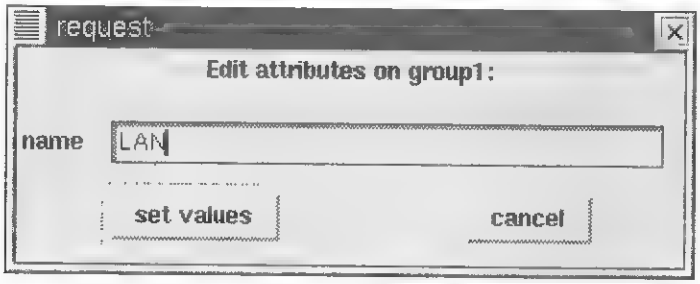


Figura 11.15 Ventana de diálogo atributos de los objetos.

- En este punto, se deberá tener una ventana que se parezca a la figura 11.16. Si no se tiene, se deben repasar los pasos anteriores para asegurarse de que se hicieron correctamente.

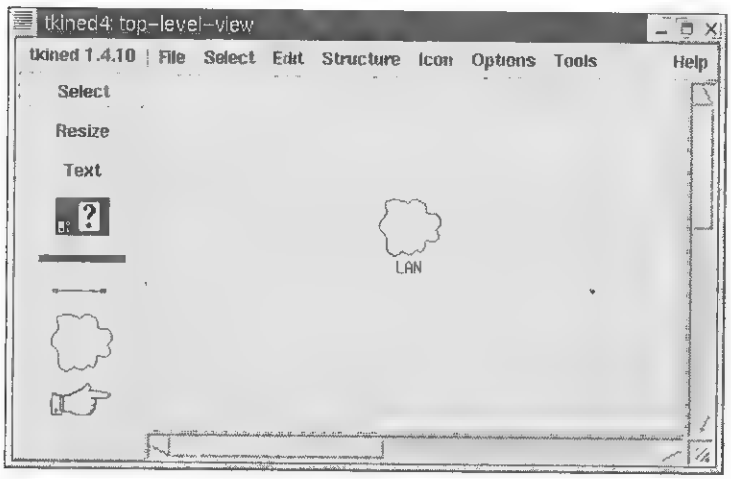


Figura 11.16 Vista lógica.

- Es importante guardar periódicamente cualquier modificación hecha a la vista. Para ello, se selecciona la opción `Save` en el menú `File`. Aparecerá un cuadro de diálogo requiriendo confirmación para actualizar la vista, como se muestra en la figura 11.17. Se da clic en el botón `Replace` para guardar los cambios más recientes. En este ejemplo, se actualizará el archivo `top-level-view`.

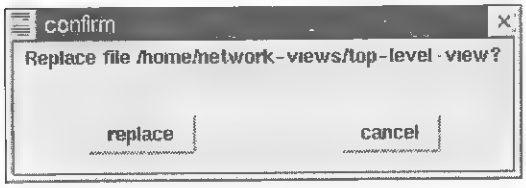


Figura 11.17 Cuadro de diálogo de confirmación.

- Se selecciona la herramienta de red. Luego, se crea una red que representará la red vertebral de la corporación. Se da un clic izquierdo al ratón, se suelta el botón, se arrastra el cursor a la derecha, y luego se da un clic izquierdo otra vez. En este ejemplo, la red vertebral incluye una serie de enrutadores Cisco interconectados a través de interfaces Gigabit Ethernet. Nótese que la herramienta no se ha actualizado con el fin de incluir iconos para los dispositivos Gigabit de trabajo en red. En cambio, se usará el icono asociado con la FDDI.
- Se cambia el tipo de objeto de red. Se selecciona la herramienta de selección. A continuación, se selecciona el icono de red que se acaba de crear, dando clic sobre el botón izquierdo del ratón. Después, se selecciona la opción `FDDI` en el menú `Icon->Network`.
- Se cambia el nombre del objeto de la red a `Backbone` modificando el elemento `Edit All Atributes...` del menú contextual. Nótese que esta ventana de diálogo presenta objetos que se pueden cambiar. Se modifica sólo el campo `name`.
- Se agregan los dispositivos de enrutador Cisco seleccionando la herramienta objeto de dispositivo y se usa el botón izquierdo para crear cuatro objetos de dispositivo. Si la red real no contiene equipo Cisco, se usan los tipos genéricos de dispositivo. Además, se altera el número de dispositivos para que coincida con su entorno de red.
- Se da clic sobre la herramienta Selection y se seleccionan todos los objetos que se acaban de crear. Se cambian los iconos predeterminados para estos dispositivos, con el fin de usar el icono Cisco, seleccionando la opción `Cisco` en el menú `Icon->Node->Network Device`.
- Se conecta cada uno de los dispositivos Cisco en la red vertebral. Se selecciona la herramienta de conexión, se da un clic izquierdo en uno de los dispositivos Cisco y luego se da un clic izquierdo en el objeto de red. Esto se repite para cada uno de los objetos Cisco.

- Se cambia la información de la dirección IP y el nombre de cada uno de los dispositivos creados sólo editando los atributos del objeto. Con los objetos de dispositivo, dos elementos de objeto se listan en el cuadro de diálogo, como se muestra en la figura 11.18. En este ejemplo, uno de los enrutadores Cisco se llama `cisco-gw1.home.com.`, con la dirección IP de `10.0.1.1`.

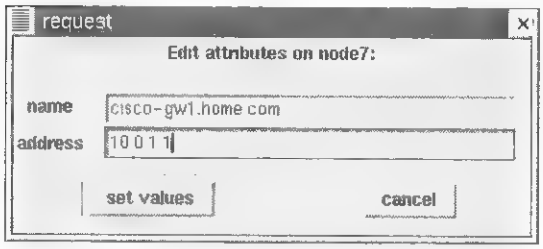


Figura 11.18 Cuadro de diálogo de atributo del objeto.

- Se mueve toda la red vertebral en el grupo LAN, se seleccionan todos los dispositivos Cisco, los objetos de red y asociados a la red, más el objeto del grupo LAN. Se selecciona la opción `Join Group` en el menú `Structure` (estructura). Una vez hecho esto, la red vertebral estará contenida en el objeto del grupo LAN.
- Para tener acceso a los objetos individuales en un grupo, se usa la opción `expand` (expandir) en el menú `Structure`. También se puede expandir el grupo seleccionando el objeto del grupo y usando el botón derecho del ratón para seleccionar `Expand group` del menú contextual.
- Una vez expandido, los elementos en el grupo se presentan en un rectángulo gris, como se muestra en la figura 11.19.
- Se puede agregar una imagen de fondo a este mapa de topología para hacer más representativa la vista. Actualmente sólo los archivos de mapa de bits X11 son soportados. Para agregar una imagen, se selecciona la opción `Import...` del menú `File` y se ingresa el nombre de archivo que contiene la imagen de fondo.
- En este ejemplo, se agregó una imagen de fondo del estado de Massachusetts a la vista de topología llamada `ma-image.xbm`, porque la compañía tiene su base en el estado. Debido a que la imagen es bastante grande, fue necesario mover la imagen completa desde la esquina superior izquierda de la ventana hasta

la esquina inferior derecha. El objeto de grupo LAN también se movió hasta la ciudad (Boston, en este caso), en la cual está localizada la oficina corporativa. Para mover el objeto, simplemente se da un clic izquierdo y derecho en el objeto, se arrastra con el ratón y se suelta el botón en la localización deseada.

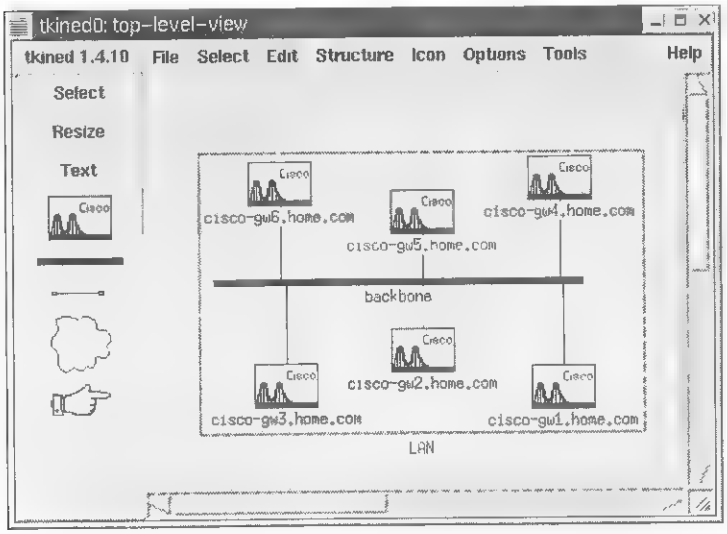


Figura 11.19 Vista de grupo.

- Se adiciona un título descriptivo para la vista. Se selecciona la herramienta de texto y se da un clic izquierdo en el área donde se quiere colocar el nuevo título. Entre los títulos posibles se incluyen: `Corporate LAN`, `Backbone Network` y `Corporate Network`, y otros similares. Si el título no está en la ubicación deseada, se mueve con el ratón, usando el mismo procedimiento analizado en el paso anterior. La figura 11.20 muestra la vista completada.

Vista WAN

Para construir una vista WAN, se siguen estos pasos:

1. Se crea otra vista `tkined` seleccionando la opción `New View` en el menú `File`.
2. Se guarda la nueva vista en un archivo llamado `WAN-view` en el directorio `/home/networkviews`.

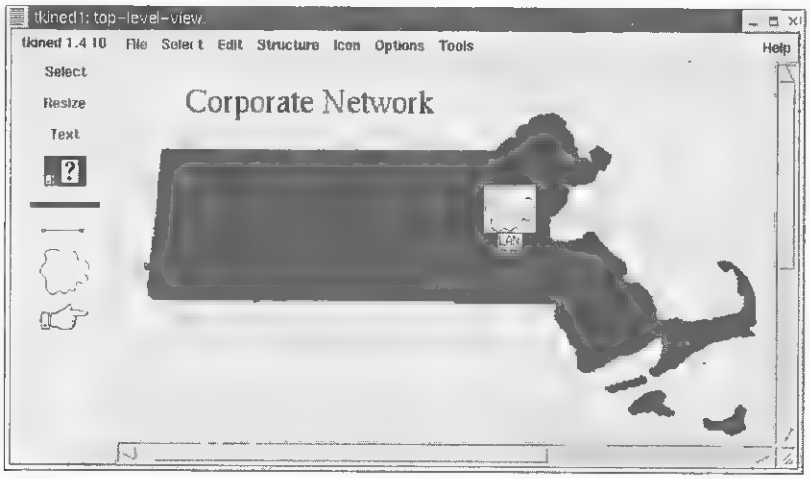


Figura 11.20 Vista LAN completada.

3. Como la compañía mantiene diferentes oficinas a lo largo y ancho del país, es razonable usar un mapa de los Estados Unidos para mostrar la localización de los dispositivos individuales que están en la WAN. Por tanto, se importa la imagen de fondo llamada `usa-image.xbm`.
4. Se crean dispositivos empleados para soportar la WAN. Estos incluyen enrutadores, módems y cualquier otro dispositivo asociado. Para este ejemplo, se ha adicionado a la vista un total de cuatro enrutadores: `wan-remote.home.com`, `fl-gw.home.com`, `tx-gw.home.com` y `ca-gw.home.com`.
5. Se crean objetos de conectividad entre los enrutadores, mostrando la topología correcta de la red.
6. Se agrega una etiqueta descriptiva usando la herramienta de texto.
7. Ahora que se ha completado esta vista, se deberá hacer un enlace desde la primera vista (`top-level-view`) hasta la `vista WAN`. Esto se puede hacer creando un objeto de referencia en la vista `top-level-view`. Para adicionar este objeto a la vista, se selecciona el icono de referencia con el botón izquierdo del ratón. A continuación, se da un clic izquierdo en el mapa de topología donde el objeto se deberá colocar.
8. Una vez creado el objeto de referencia, se debe agregar a la vista `WAN`. Para hacer esto, se da un clic derecho en el objeto

referencia para presentar el cuadro de diálogo que muestra en la figura 11.21. Se introduce el nombre `WAN-view` en el campo `name`, para que sirva como nombre del objeto. A continuación, se especifica el nombre de ruta y el nombre de archivo completo del archivo que contiene el mapa de topología `WAN-view`. En este ejemplo, la ruta y el archivo es `/home/network-views/wan-view`. Se da un clic en el botón `set values` para guardar esta información.

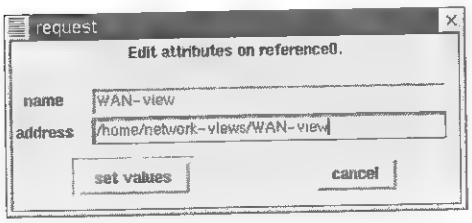


Figura 11.21 Ventana de diálogo de los atributos del objeto referencia.

9. Para asegurarse de que el enlace del objeto referencia es operacional, se da un clic derecho en el enlace del objeto, y cuando la ventana contextual aparezca, se selecciona el elemento `Open (this view)`. Si el enlace está funcionando, un mapa de topología de la vista `WAN` deberá remplazar la vista existente, como se muestra en la figura 11.22.

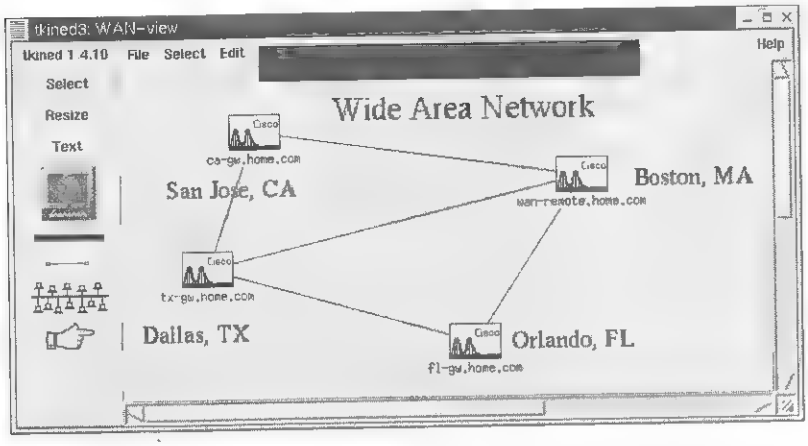


Figura 11.22 Vista de topología WAN.

Uso de las imágenes de fondo

Las imágenes que representan muchos estados de Estados Unidos están localizadas en el directorio `Images` en el nivel superior del CD-ROM. Estas imágenes se pueden usar como fondos en las vistas de topología **tkined**. Bajo este directorio, un archivo `README.txt` contiene información adicional de cómo recuperar y usar estas imágenes.

Administración de la red

En este punto, las vistas de topología de la red ya se han creado, sea a través del proceso de descubrimiento automatizado, o manualmente. En cualquier caso, se supone que los elementos de la red, como los enrutadores y los conmutadores de red, se han creado y se puede empezar a usar ahora algunas de las características poderosas de **tkined** para administrar estos dispositivos individuales.

Con el menú `Tools`, presentado en la figura 11.23, se puede tener acceso a muchas de las capacidades de administración de dispositivos. Este menú suministra acceso a un gran conjunto de funciones de administración.

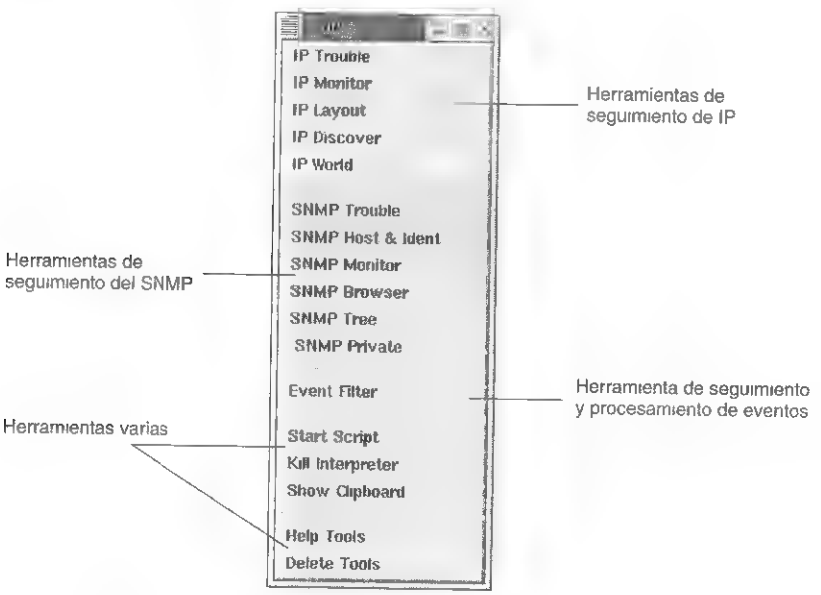


Figura 11.23 Menú `Tools`.

Las tareas que se encuentran en el menú `Tools` se pueden dividir en las siguientes categorías:

- Seguimiento del protocolo de Internet (IP)
- Seguimiento del SNMP
- Seguimiento y procesamiento de eventos
- Tareas varias

La mayoría de las opciones, cuando se seleccionan, crean un nuevo menú dinámico a la derecha del menú `Tools`. Por ejemplo, seleccionar la opción `IP-Trouble` añade un nuevo menú, `IP-Trouble`, a la barra del menú principal, como se muestra en la figura 11.24. Sin embargo la remoción del menú no es automática; el usuario debe borrar el menú manualmente seleccionando la última opción del menú `Tools`, `Delete IP-Monitor`.

Figura 11.24 Adición de un nuevo menú en el menú principal.

El menú `IP Monitoring` incluye los cinco submenús que se enumeran en la tabla 11.1. Algunas de las funciones de seguimiento usan las facilidades de las llamadas a procedimientos remotos (RPC) de los dispositivos objetivo, para ganar información de administración. Por tanto, esta facilidad debe estar activa para que estas funciones de seguimiento funcionen correctamente.

Muchas de estas funciones también requieren una selección de uno o más sistemas objetivo para efectuar una tarea específica. Por tanto, las funciones seleccionadas de los menús no se ejecutarán sin una selección objetivo.

Submenú IP Trouble

El submenú `IP-Trouble` contiene una gran cantidad de funciones que ayudan a diagnosticar problemas del sistema y de conectividad específicos. Ésta es la fachada de muchos de los comandos UNIX comunes del sistema y de la red, como **ping**, **traceroute**, **telnet**, **finger** y **whois**. La figura 11.25 muestra el submenú como una ventana autónoma. Todos los menús disponibles bajo **tkined** se

Tabla 11.1 Submenús del menú de seguimiento IP.

Submenú	Descripción
`IP Trouble`	Herramientas utilizadas para diagnosticar problemas específicos de conectividad o relacionados con el sistema
`IP Monitoring`	Herramientas usadas para hacer seguimiento a servicios específicos
`IP Layout`	Opciones para controlar el diseño de una o más vistas de topología
`IP Discover`	Herramienta usada para descubrir todos los dispositivos y sistemas en una red dada
`IP World`	Facilidad visual de *traceroute*

pueden "arrancar" y acceder a través una ventana regular. *Véase* la sección "Referencia de menús" para obtener instrucciones de cómo activar un menú para arrancar.

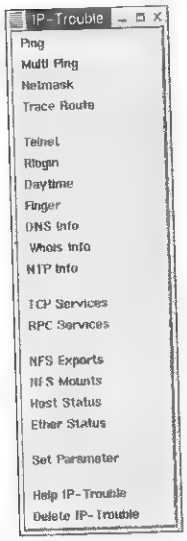


Figura 11.25 Submenú `IP-Trouble`.

Las opciones listadas en este submenú se describen a continuación.

Ping

Esta opción implementa los servicios ping estándar para determinar la conectividad general de la red. El capítulo 4, "Herramientas y utilidades centrales del sistema", analiza este comando. Éste presenta una ventana de resultado que muestra el tiempo de ida y vuelta para la respuesta ICMP (ping), cuando el comando es exitoso. Si la tarea falla, un mensaje `not reachable` (no alcanzable) se presenta en su lugar.

Multi Ping

Esta opción suministra la capacidad de alterar algunas de las opciones ping cuando revisa la conectividad general de la red. Invocar esta función hace que la aplicación presente la ventana de diálogo que se muestra en la figura 11.26. Esta ventana contiene dos elementos de desplazamiento que se pueden usar para variar el número de bytes y la cantidad máxima de bytes que se envía con cada solicitud ICMP. El número de bytes enviados determina el número de solicitudes ICMP que se transmiten.

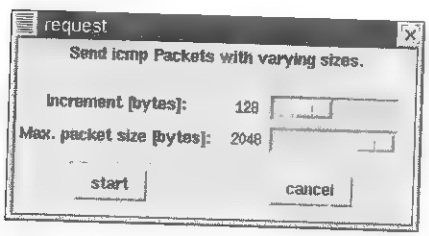


Figura 11.26 Ventana de diálogo Multi Ping.

Se da un clic al botón `Start` para empezar a emitir las solicitudes ICMP. Al usar los valores predeterminados, se envía un total de 17 solicitudes, como se muestra en la ventana de resultados que se presenta en la figura 11.27. Esta herramienta es más útil para obtener información genérica de rendimiento de uno o más dispositivos de red, de una manera más automatizada.

Netmask

Esta opción presenta la máscara definida de subred empleada por el *host* objetivo. Acerca del éxito, ésta presenta un mensaje

similar a: `Netmask for monet.home.com [10.0.2.27] 255.255.255.0` para el *host* `monet`, indicando el uso de `255.255.255.0` (24 bits) para la máscara de la subred. Si la herramienta no puede obtener esta información, se presenta una lista vacía de máscara de `0.0.0.0`.

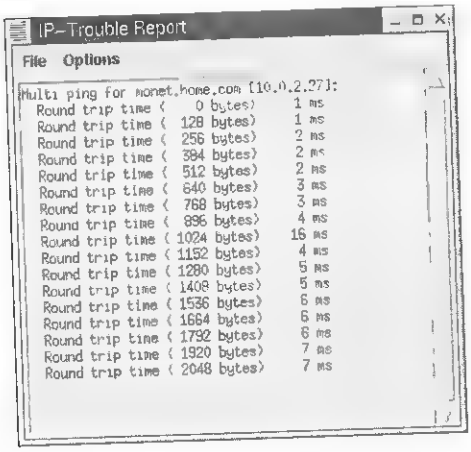


Figura 11.27 Ventana de resultado Ping.

Traceroute

Esta opción implementa los servicios estándar de traceroute para determinar el camino de enrutado entre el sistema local que está corriendo **tkined** y el sistema remoto objetivo. Éste presenta la misma salida en el mismo formato del comando estándar **traceroute**. El capítulo 4,"Herramientas y utilidades centrales del sistema", analiza el comando **traceroute**.

Telnet y Rlogin

Esta opción suministra un acceso genérico remoto al sistema objetivo invocando los servicios **telnet** o **rlogin**. Cuando se selecciona cualquiera de ellos, presentan una nueva ventana con un indicador de comandos de inicio de sesión. Nótese que si se han seleccionado dos o más *hosts* objetivo, se creará un número apropiado de ventanas terminales.

Daytime, Finger, DNS Info, Whois y NTP Info

Cada una de estas opciones obtiene información miscelánea acerca del *host* objetivo y presenta una ventana de resultado que contiene la información o cualquier mensaje de error.

Servicios TCP/RPC

Estas dos opciones intentan descubrir cuáles servicios, puertos TCP y/o llamadas a procedimientos remotos (RPC), están corriendo en el objetivo remoto. En el caso de los servicios TCP, la salida es similar al comando **nmap** que se explicó en el capítulo 5. Esta opción recuperará todos los servicios RPC que están corriendo actualmente. El sondeo de un sistema servidor Sun llamado `bubbles` obtuvo la salida que se muestra en la figura 11.28.

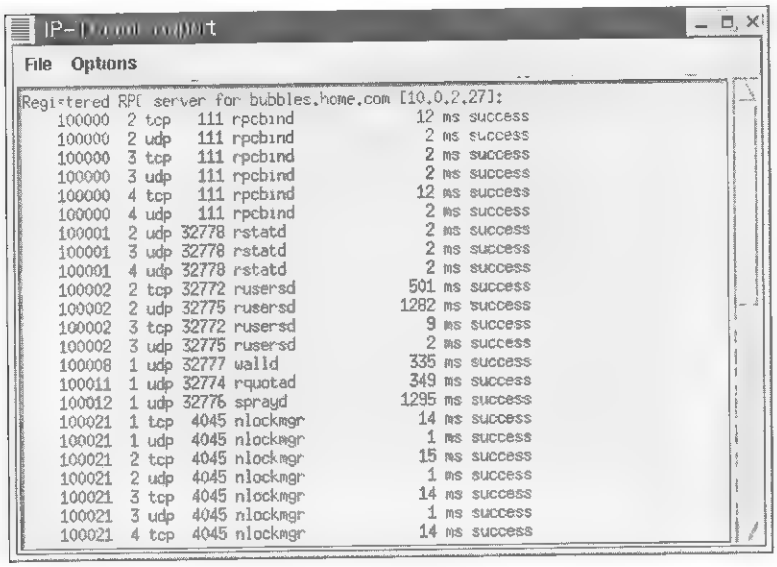


Figura 11.28 Listado de los servicios RPC.

NFS Exports/NFS Mounts

Estas opciones listan cualquier sistema de archivos exportado o sistema de archivos montado en el objetivo remoto.

Host status

Esta opción suministra la capacidad de ejecutar un comando **vmstat** en un *host* objetivo remoto a través del proceso **rstatd**. Este comando suministra también información de cuánto tiempo ha estado el dispositivo activo. Cuando se ejecuta, presenta una ventana de diálogo que permite al usuario alterar el intervalo de sondeo. Este intervalo controla la cantidad de tiempo entre ciclos de consulta. La figura 11.29 muestra esta ventana.

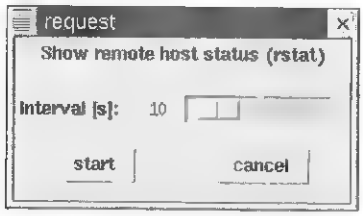


Figura 11.29 Ventana de diálogo de intervalo.

La figura 11.30 muestra un ejemplo de la salida que se genera al ejecutar esta tarea.

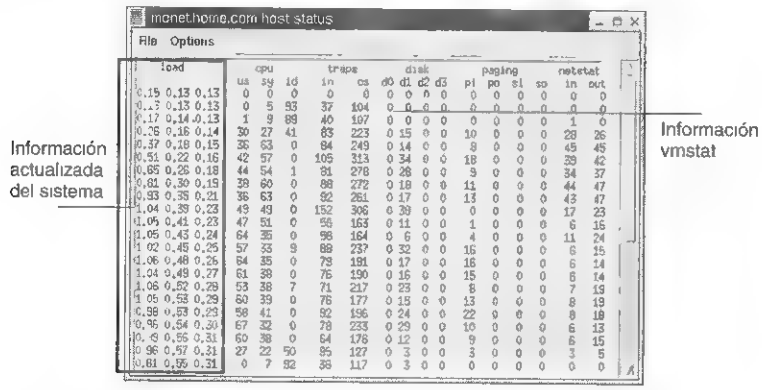


Figura 11.30 Ventana de resultado remoto.

Ether Status

Esta opción suministra la capacidad de obtener estadísticas parecidas a las Ethernet de un sistema objetivo remoto, a través del proceso `etherd`. Al igual que la función `Host Status`, ésta presenta una ventana de diálogo que permite al usuario alterar el intervalo de sondeo.

Set Parameter

Esta opción presenta una ventana de diálogo que permite al usuario cambiar algunos parámetros básicos asociados con las opciones contenidas en el menú `IP-Trouble`.

La ventana contiene cuatro elementos; los primeros tres controlan las características generales operativas del comando **ping**, mientras que la otra controla el límite de cantidad de saltos utilizados con el comando **traceroute**.

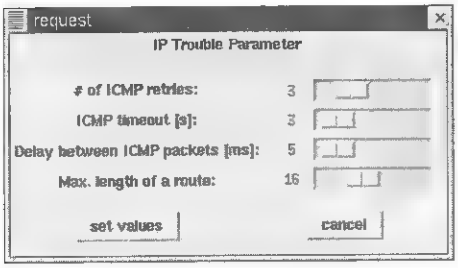


Figura 11.31 Ventana de parámetro IP-Trouble.

Help IP-Trouble

Esta opción presenta una nueva ventana que contiene ayuda genérica en todas las opciones del menú `IP-Trouble`. Esto es útil para buscar los menús de opciones específicas, pero no suministra detalles específicos respecto a cada opción.

Delete IP-Trouble

Esta opción remueve el menú `IP-Trouble` de la ventana principal.

Menú IP-Monitoring

El menú `IP-Monitoring`, que se muestra en la figura 11.32, suministra tareas de seguimiento de fallas relacionadas con el comportamiento de rendimiento y la conectividad de red de un sistema UNIX. Algunas de las opciones del menú crean gráficos de rendimiento en el panel de topología. Como se verá, estos gráficos son una excelente forma de hacer seguimiento a información específica de forma continua.

Las opciones listadas en el menú `IP-Monitoring` se describen a continuación.

Check Reachability

Esta opción hace seguimiento a la accesibilidad del dispositivo de la red y suministra una forma diferente de indicar cuándo ocurren los problemas. Para habilitar esta característica, simplemente se seleccionan los dispositivos objetivo a los que debería hacérseles seguimiento, y se selecciona la opción `Check Reachability`. Cuando la conectividad de la red se interrumpe, debido a un problema de la red o de un dispositivo, el icono de dispositivo de topología asociado que representa al sistema al que se le hace seguimiento brillará en forma intermitente y se tornará rojo. La figura 11.33 muestra una vista de topología con

un dispositivo (llamado `route-1.home.com`) que falló en la prueba de accesibilidad. Sus iconos han cambiado para indicar que ha ocurrido este problema.

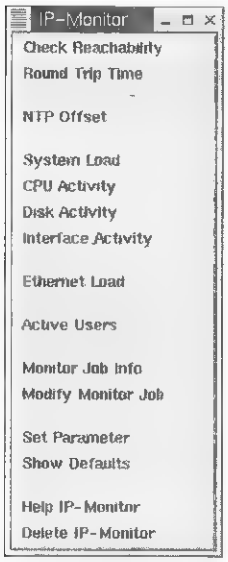


Figura 11.32 Menú IP-Monitoring.

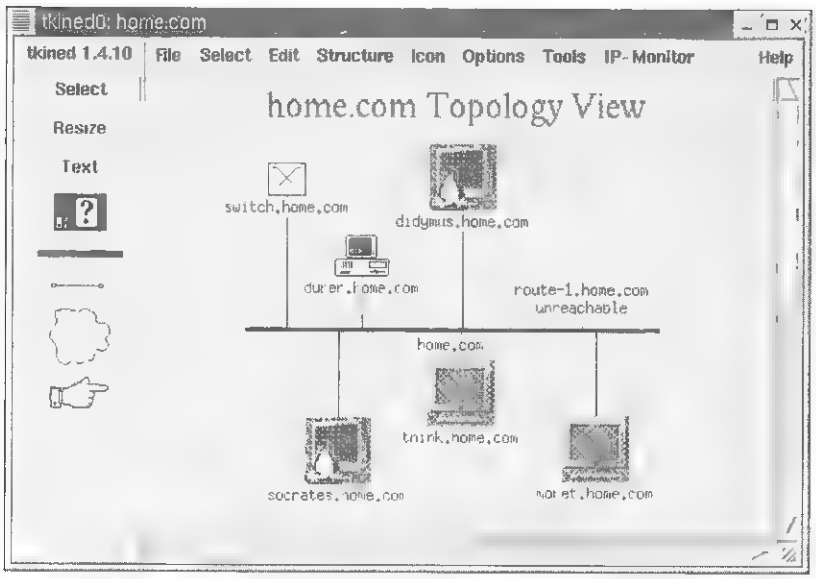


Figura 11.33 Cambio de un icono de dispositivo para denotar una falla.

En la figura, el icono aparece blanco. Esto se debe a que el icono está brillando de manera intermitente entre los colores blanco y rojo de manera continua; cuando la vista instantánea de la pantalla de la ventana se tomó, el icono era blanco en ese momento. Cuando la conectividad se restaura en el sistema, el icono retorna a la normalidad. Esto suministra un mecanismo visual para alertar acerca de cuándo un dispositivo o sistema particular está abajo (*down*).

Téngase en cuenta que factores no relacionados con un sistema pueden causar que éste parezca estar abajo. Por ejemplo, en muchas redes, un conmutador u otro dispositivo de trabajo en red suministra conectividad entre los sistemas objetivo a los que se les hace seguimiento y el *host* que está corriendo el software de seguimiento. Por ejemplo, considérese la red que se muestra en la figura 11.34.

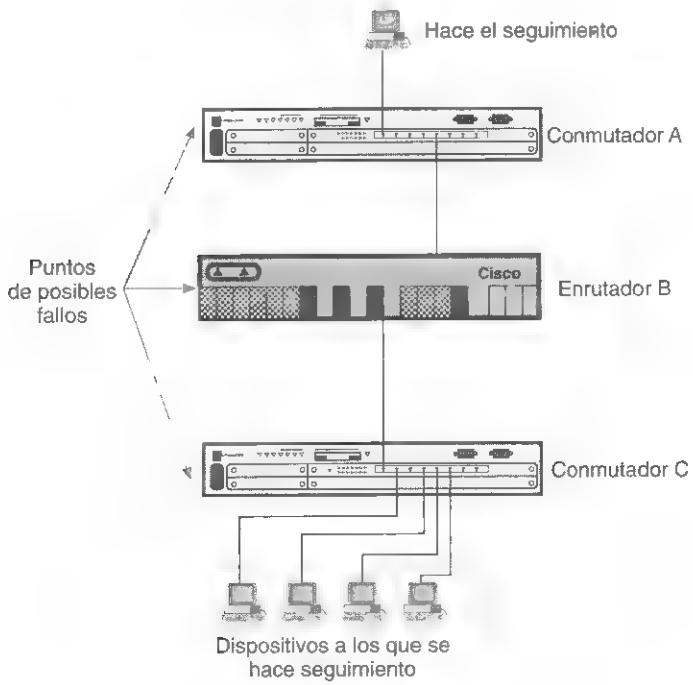


Figura 11.34 Fallas posibles de seguimiento.

Esta red consta de tres dispositivos principales y dos conmutadores 3Com LAN interconectados a través de un enrutador Cisco. Si el `enrutador B` o el `conmutador C` falla por cualquier razón, la conectividad entre los sistemas a los que se les hace seguimiento y los dispositivos de seguimiento aparecerán como si estuvieran abajo (inactivo). En realidad, un dispositivo intermedio es el que ha causado este problema. ¿La solución? Se planea hacer seguimiento no sólo a los sistemas y otras estaciones de trabajo importantes. Tómese en consideración toda la ruta entre los sistemas críticos para la red que suministrará los servicios principales de seguimiento.

Para avanzar un poco más, considérese el ejemplo donde a todos los dispositivos se les está haciendo seguimiento y la energía suministrada en el `Conmutador C` decide tomar unas vacaciones y volverse inoperante. En este punto, si se supone que el conmutador no contiene suministros de energía redundantes, el conmutador está condenado al fracaso. Desde la perspectiva de **tkined**, los dispositivos conectados a este conmutador y el conmutador en sí han quedado desconectados del resto de la red. La vista de topología contiene ahora un montón de iconos brillando de manera intermitente. El problema ahora es, ¿cuál(es) dispositivo(s) está(n) realmente abajo?

La respuesta a esta pregunta, desafortunadamente, requiere algún conocimiento acerca de la topología de la red. Un técnico de red experimentado puede examinar la topología y determinar el mejor lugar para comenzar a buscar la causa más probable. El primer paso en una situación como ésta es empezar a aislar el problema. En algún grado, la herramienta **tkined** ya ha hecho esto; ésta muestra cuál parte de la red está experimentando problemas. Será importante examinar el problema en etapas, asegurándose de la conectividad a lo largo del camino. Por ejemplo, si se ve que un conmutador en particular está abajo, se debe intentar establecer la accesibilidad al dispositivo más cercano al conmutador. Se tiene que seguir sondeando cada vecino del sistema afectado hasta que se haya determinado la verdadera causa, la cual, en este caso, requiere el remplazo del suministro de energía que falló.

Round-Trip Time

Esta opción produce un gráfico reducido que muestra la demora de ida y vuelta usando el protocolo de mensajes de control en Internet (ICMP). Lo interesante sobre esta característica es que el gráfico se agrega el panel de topología junto al icono del dispositivo. En esencia, esta herramienta suministra la capacidad de

visualizar el comportamiento del sistema y de la ruta de red asociada al sistema. Esto significa que se pueden crear vistas específicas para presentar información relacionada con el rendimiento. Considérese la vista presentada en la figura 11.35, la cual muestra dos servidores, cada uno con un gráfico asociado de tiempo de ida y vuelta (RTT). Todo marcha bien en ambos sistemas porque el RTT es un milisegundo. Esto indica que las rutas de la red entre el sistema que está corriendo **tkined** y los *hosts* objetivo están libres de congestiones u otros problemas que podrían afectar el rendimiento de la red.

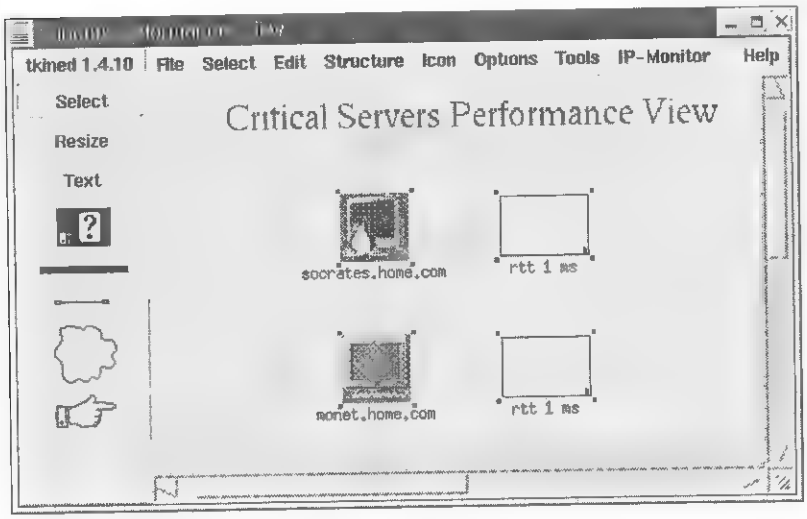


Figura 11.35 Vista de rendimiento específico.

System load, CPU, Disk e Interface Activity

Cada una de estas opciones suministra capacidades de seguimiento, como el gráfico de tiempo de ida y vuelta anterior. Cuando se selecciona, cada una creará un gráfico reducido, como se muestra en la figura 11.36. En esta figura, se creó una nueva vista específica para un solo sistema, llamado `monet`. La vista contiene cinco gráficos diferentes que indican la demora de ida y vuelta, los promedios de carga del sistema, la actividad de la CPU, el rendimiento de la interfaz de red y la actividad del disco.

Para entender la información de rendimiento presentada en los gráficos se requiere conocer la salida de ciertos comandos de seguimiento UNIX. Está fuera del alcance de este libro examinar

cada una de las herramientas de seguimiento de administración del sistema. Por tanto, el lector deberá investigar **vmstat, rstatd, uptime** y los comandos relacionados, para obtener información adicional.

El diseño de la vista no es crítico y, con un poco de imaginación, las vistas personalizadas se pueden colocar juntas, lo que las hace al mismo tiempo funcionales y fáciles de usar. Si muchos sistemas requieren la cantidad de seguimiento mostrado en la figura 11.36, se recomienda que cada una de las vistas del sistema se cree de forma separada, e incluso enlazada con su propio icono de referencia. Como otra alternativa, se pueden usar los elementos del grupo en lugar de los iconos de referencia; la decisión realmente es del usuario.

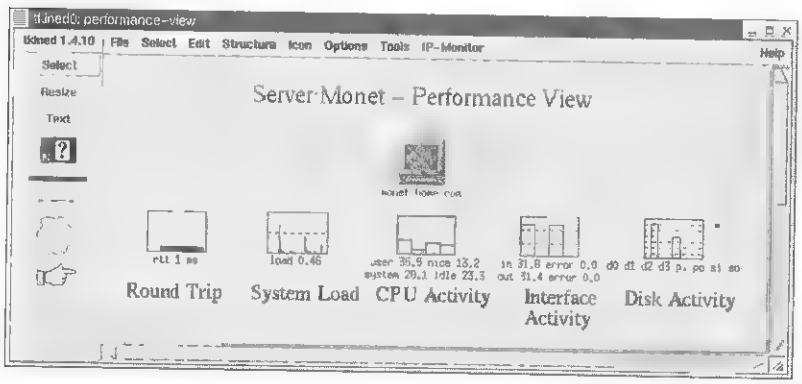


Figura 11.36 Vista de rendimiento de `monet`.

Ethernet Load/Active Users

La opción `Ethernet Load` muestra la carga de la red de la interfaz predeterminada, tal como la reporta el proceso etherd. La opción `Active Users` suministra una lista de todos los usuarios registrados actualmente en un sistema particular.

Monitor Job Info/Modify Monitor Job

Estas opciones suministran la capacidad de hacer seguimiento y alterar los procesos de seguimiento **tkined**. Se puede listar cada una de las tareas de seguimiento que están corriendo actualmente, ejecutadas por **tkined**, usando cualquiera de estas opciones. La opción `Monitor Job Info` simplemente lista todos los procesos activos, mientras que la opción `Modify Monitor Job` permite al usuario alterar ciertos parámetros operativos o incluso matar un

proceso en particular. La figura 11.37 muestra la ventana que se presenta cuando se selecciona la opción `Modify Monitor Job`.

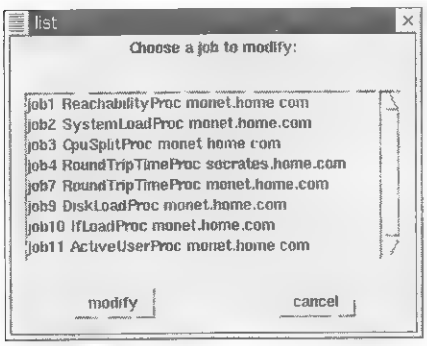


Figura 11.37 Ventana de selección `Modify Job`.

Esta ventana contiene una lista de los trabajos a los que se les ha hecho seguimiento, desde la cual el usuario puede seleccionar un solo trabajo para revisarlo o modificarlo. Para modificar un trabajo en particular, se selecciona, y se pulsa el botón `Modify`. Como resultado, se presentará una nueva ventana, como se muestra en la figura 11.38. Esta ventana controla el intervalo de seguimiento, el estado del trabajo, los valores de umbrales ascendente y descendente y las acciones de umbral, y suministra la capacidad de parar las tareas a las que se les ha hecho seguimiento. Si se quiere parar el trabajo, se usa el botón `Kill Job`.

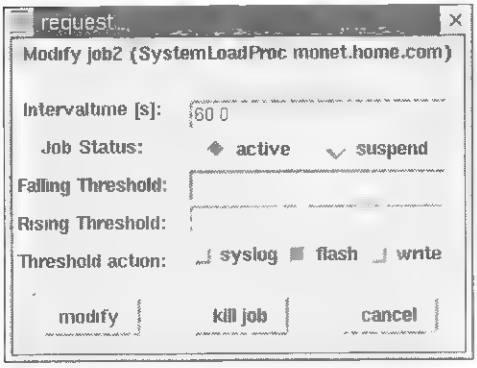


Figura 11.38 Ventana de la tarea de seguimiento `Modify`.

Set Parameter

Esta opción suministra la misma función que la opción de menú `IP-Trouble`, pero contiene un conmutador adicional alterno que controla la creación de diagramas de gráficos. El valor predeterminado es `false`.

Delete IP-Monitor

Esta opción borra el menú `IP-Monitor`. Sin embargo, si los trabajos de seguimiento están activos, al seleccionar esta opción de menú se presentará un cuadro de diálogo que informa al usuario qué tareas pendientes están aún operando. Borrar el menú significa que cualquier trabajo al que se le hace seguimiento se detendrá. Al usuario se le da la opción de matar esas tareas y salir del menú o cancelar la tarea de borrar todo el menú al mismo tiempo. Borrar el menú `IP-Monitor` remueve automáticamente ciertos gráficos, como los de actividad de la interfaz y actividad del disco, de la vista de topología.

Menú IP-Layout

El menú `IP-Layout` contiene funciones que controlan cómo se organizará una vista de topología. Las opciones en este menú son buenas para manejar situaciones donde se ejecutó un proceso de descubrimiento IP, y donde se ha creado una gran cantidad de elementos de dispositivos de red. Este menú suministra una forma de automatizar cómo los dispositivos son organizados y situados en la vista. La figura 11.39 muestra el menú `IP-Layout`.

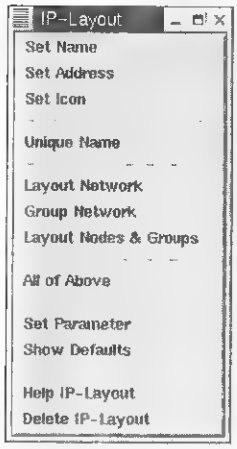


Figura 11.39 Menú `IP-Layout`.

Las opciones listadas en el menú `IP-Layout` se describen enseguida.

Set Name, Address e Icon

Estas opciones del menú causan que lo que se muestra en un área de la etiqueta del objeto incluya el nombre del objeto, la dirección o el nombre del icono. Por ejemplo, para hacer que cada dispositivo contenga la dirección en la etiqueta, se selecciona la opción del menú `Set Address`.

Unique Name

Esta opción retirará las extensiones del nombre de dominio para cada elemento de icono seleccionado. El propósito de la opción es acortar el nombre de cada objeto, de tal forma que éste incluya sólo la porción del nombre del *host*, no el nombre de dominio.

Layout Network, Group Network y Layout Nodes & Groups

Estas opciones de menú suministran tareas automatizadas para ayudar en la organización de una vista de topología que contiene muchos objetos. También se pueden usar con una pequeña cantidad de dispositivos, pero puede ser más fácil organizar esto usando un enfoque manual, particularmente si la vista va a contener un aspecto personalizado. La opción `Layout Network` posicionará todos los nodos que están adjuntos a una red alrededor de esa red, de manera ordenada. La opción `Group Network` hará que todos los miembros de una red dada sean miembros de un grupo, los cuales serán nombrados después de la dirección de la red. La opción `Layout Network` intenta controlar la organización de grupos y de objetos de dispositivos individuales juntos en una vista. Desafortunadamente, la organización de estos objetos no es lo que uno esperaría. Esta función organiza grupos de una manera razonable, pero coloca los objetos de dispositivo uno encima del otro, lo cual no es del todo útil. Es recomendable colocar los objetos individualmente o usando una de las otras opciones automatizadas de organización del menú.

All of Above

Esta opción parece sugerir que ejecutará cada uno de los elementos del menú que están arriba de ella. Cuando se selecciona, esta opción empieza a cambiar los nombres de los elementos de objeto al nombre completo. La aplicación **tkined** se cayó después de esto. Aun después de intentos repetidos, la opción no completó

la tarea o hizo que la aplicación se detuviera. Por tanto, se recomienda no usar esta opción del menú.

Show Parameter

Esta opción presenta una nueva ventana que contiene parámetros de menú de organización que el usuario puede alterar. En la práctica, no se necesitará cambiar los valores predeterminados, y se puede ignorar esta opción.

Delete IP-Layout

Esta opción cierra el menú `IP-Layout`. Ejecutar esta tarea no afecta ninguna de las tareas de organización ejecutadas previamente, elementos de objeto existentes o vistas de topología asociadas.

Menú IP-Discover

El menú `IP-Discover` automatiza el proceso de recolección e inventario de los dispositivos de la red, con el propósito de construir una o más vistas de topología. El proceso de conducir un descubrimiento se estudia en la sección "Descubrimiento de la red", al comienzo de este capítulo. Aquí se justifica una nota de precaución, porque un proceso de descubrimiento de la red puede afectar adversamente el rendimiento de la red, si se usa incorrectamente o muy seguido. El seguimiento de las siguientes reglas cuando se usa el proceso automatizado de descubrimiento ayudará a minimizar los problemas:

- Se deben descubrir sólo aquellas redes con las que se trabaja realmente o se tiene el permiso de los dueños de la red para hacer inventario.
- Nunca se debe dejar activo el proceso de descubrimiento sin alguien que le haga seguimiento a éste. Debido a que el proceso de inventariar grandes redes es una tarea que consume tiempo, ésta puede tomar varios días para descubrir completamente la red. Puede no ser una buena idea iniciar el proceso e irse. Si ocurre un problema que afecta adversamente el resto de la red, es mejor conocer el problema antes que la comunidad de usuarios (o el jefe).
- Cuando se hace inventario en una red grande, es mejor descubrir la red en porciones pequeñas, que intentar hacer todo y dejar que el proceso abarque varios días. El argumento más importante contra los procesos de descubrimiento de gran magnitud es que éstos tienden a causar problemas relaciona-

dos con la red y el software podría caerse completamente o sondear la red de manera continua, como resultado de un *bug* del software u otro problema.

- Se debe estar informado acerca de la naturaleza exacta del proceso de descubrimiento. Es importante saber cuáles objetos SNMP MIB y otros elementos son consultados, para hacer un inventario de la red.

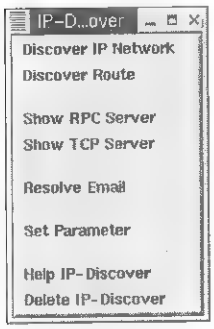


Figura 11.40 Menú `IP-Discover`.

Las opciones listadas en el menú `IP-Discover` se describen enseguida.

Discover IP Network

Como se muestra en la sección "Descubrimiento de la red", esta opción invoca el proceso de descubrimiento para una red basada en IP. Desafortunadamente, como se ha dicho, **tkined** no soporta otros protocolos de capa de red, como IPX. Al seleccionar la opción `IP Discover` se presentará una nueva ventana de diálogo. Esta nueva ventana contiene un cuadro de texto que permite al usuario añadir una o más direcciones de destino de red a descubrir.

El proceso de descubrimiento usa tanto el ICMP como el SNMP para inventariar y consultar dispositivos en la red.

Discover Route

Esta opción del menú conducirá al descubrimiento de la ruta hacia un *host* destino u otro dispositivo. Cuando se establece la ruta a un destino remoto, se crea una serie de enrutadores (puertas de enlace) e iconos de red que representan la ruta de la red desde el sistema local hasta el sistema remoto. Este tipo de descubrimiento es más útil cuando se intenta aprender más acerca de los nodos individuales que se usan cuando se alcanza un sistema remoto.

Show RPC/TCP Server

Estas dos opciones intentan sondear un servicio de destino remoto RPC o TCP, para ver si actualmente está operando y aceptando conexiones de servicio. Cuando se seleccionan, cada una presenta un cuadro de diálogo que contiene una lista de procesos de servidor remoto para consultar. Al seleccionar la opción RPC, se presenta una lista de servidores, como se muestra en la figura 11.41.

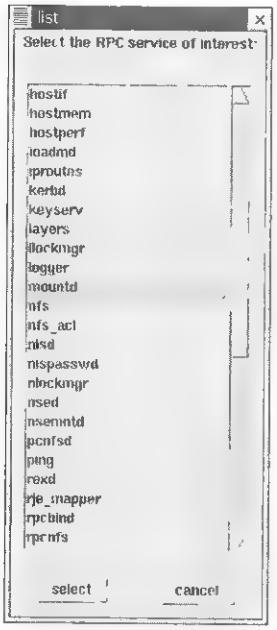


Figura 11.41 Lista de consulta de servidor RPC.

En la figura se seleccionó el proceso mountd. Este proceso se usa cuando el montaje remoto de un sistema de archivos se configura en un sistema. El propósito principal de estas opciones del menú es intentar el descubrimiento de una pequeña clase de los dispositivos y sistemas. Por ejemplo, es posible investigar todos los sistemas que suministran capacidades de montaje remoto de un sistema de archivos. La lista de servicios remotos que se muestra en la ventana de diálogo representa muchos de los servicios estándar para la mayoría de los sistemas UNIX. Por tanto, no debería ser difícil hacer un inventario de la red con casi cualquier servicio que se necesite.

Resolve Email

Esta opción del menú proporciona un mecanismo para determinar la localización de sistemas de correo electrónico que suministran un buzón para un usuario de correo electrónico dado. Cuando se selecciona, la opción presenta un cuadro de diálogo que permite al usuario agregar una dirección de correo electrónico en un campo de texto. Esta función se usa principalmente para identificar cuáles sistemas se usan para soportar el correo electrónico y para hacer seguimiento a estos sistemas específicos.

Set Parameter

Esta opción controla parámetros genéricos operacionales para las opciones del menú de descubrimiento. En la práctica, no se necesitará cambiar los valores predeterminados, y se puede ignorar en gran medida esta opción.

IP World Menu

El menú `IP World` suministra una característica visual traceroute, que cuando se le da un destino, registra los saltos de ruta en un mapa mundi. Puesto que este menú proporciona los mismos servicios básicos que el comando **xtraceroute** que se analizó en el capítulo 5, "Utilidades y herramientas adicionales del sistema", esta opción no se describe aquí.

Menú SNMP

Las herramientas relacionadas con el SNMP suministran una forma de consultar información específica SNMP y de hacer seguimiento a objetos MIB. En general, estos servicios equivalen a tener una forma de interrogar los agentes SNMP de una manera automatizada.

Menú SNMP-Trouble

El menú `SNMP-Trouble`, que se muestra en la figura 11.42, suministra opciones para consultar los dispositivos por información específica. Cuando se presentan problemas específicos, se usa este menú para aprender más acerca del problema e implementar una solución.

Las opciones en el menú `SNMP-Trouble` se describen enseguida.

SNMP Devices

Esta opción suministra una prueba genérica para determinar cuál de los dispositivos seleccionados en el panel de topología responde a solicitudes SNMP. El objeto `system.SysDescr.0` MIB-II se

consulta, como resultado de la selección de esta opción. La opción se suministra como una forma rápida de establecer conectividad básica SNMP a los dispositivos de red.

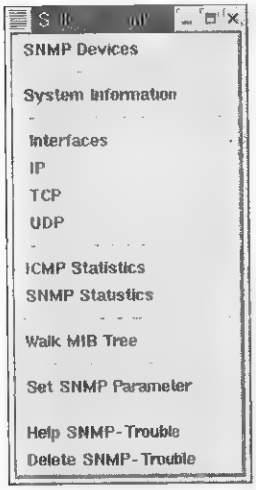


Figura 11.42 Menú `SNMP-Trouble`.

System Information

Esta opción muestra información genérica SNMP acerca del sistema objetivo. Suministra el equivalente a un recorrido (*walk*) en el grupo `system` MIB-II, como se muestra en la figura 11.43.

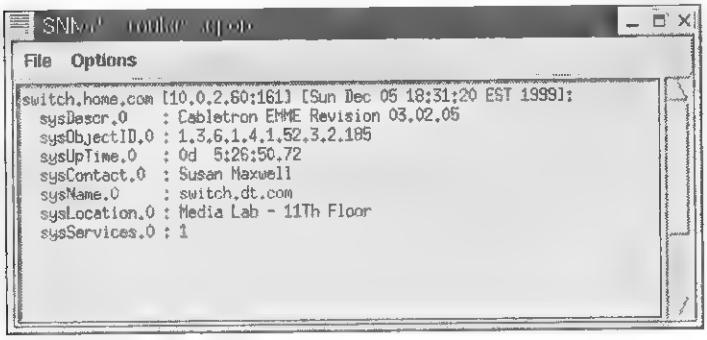


Figura 11.43 Ventana de presentación de la información del sistema.

Interfaces, IP, TCP y UDP

Estas opciones del menú suministran acceso a los objetos asociados con el grupo MIB-II. Cada opción contiene una o más subopciones para presentar tablas específicas MIB o información de grupo.

ICMP y SNMP Statistics

Como las anteriores, estas opciones de menú suministran acceso a objetos adicionales MIB-II.

Walk MIB Tree

Al igual que el comando **snmpwalk** UCD, esta opción de menú suministra una forma automatizada de recorrer el árbol MIB. Dado un grupo MIB, ésta recorre todo el árbol MIB hasta que se detecta el final del MIB.

Set SNMP parameter

Esta opción del menú permite al usuario alterar algunos de los parámetros predeterminados SNMP compartidos por todas las funciones del menú. Al seleccionar esta opción, se presenta la ventana que se muestra en la figura 11.44.

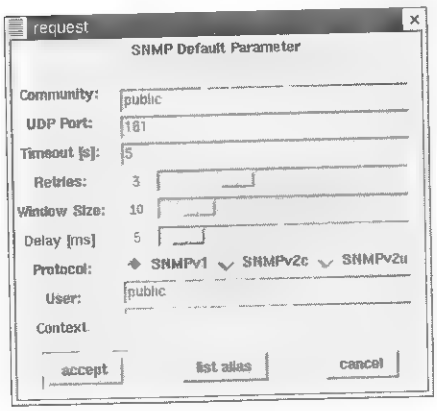


Figura 11.44 Ventana de opciones de parámetros SNMP.

La ventana contiene algunos de los parámetros básicos SNMP, como nombre de comunidad, puerto SNMP (UDP), valores de extemporaneidad, y así sucesivamente. Si la red usa cadenas de comunidad no estándar (lo que se recomienda bastante), éste sería el lugar para introducir el valor apropiado.

Menú SNMP-Host

El menú `SNMP-Host` se usa para recuperar objetos específicos de la MIB RECURSO DE HOST. Esta MIB suministra objetos específicos relacionados con el almacenamiento, los sistemas de archivos y la información del proceso. Puesto que una descripción de esta MIB está más allá del alcance de este libro, este menú no se describirá.

Menú SNMP-Monitor

Este menú suministra acceso a las opciones para el seguimiento de objetos MIB específicos. La figura 11.45 muestra el menú `SNMP-Monitor`. El propósito básico de este menú es suministrar un seguimiento continuo de los elementos de dispositivo para determinar condiciones anormales u otros problemas.

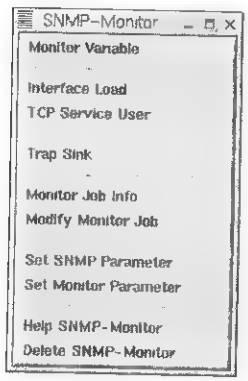


Figura 11.45 Menú `SNMP-Monitor`.

Las opciones del menú `SNMP-Monitor` se describen a continuación.

Monitor Variable

Una de las tareas más importantes que ofrece **tkined** es la capacidad de hacer seguimiento a uno o más objetos MIB específicos. Al seleccionar la opción `Monitor Variable`, se presenta una ventana de diálogo que permite al usuario introducir un nombre de objeto MIB, como se muestra en la figura 11.46.

Se puede agregar cualquier nombre válido de objeto MIB en este cuadro de texto. Al seleccionar el botón `start`, se crea un gráfico en la vista de topología, con los valores de este objeto. En

la figura 11.46 se introdujo el objeto MIB-II `ifInOctects.4`. Recuérdese que en el capítulo 6, "Vista general de MIB-II", se analizó cómo este objeto contiene el número total de bytes entrantes para la quinta interfaz en el sistema. Una vez que **tkined** comienza a hacer seguimiento a este objeto, se puede mantener un gráfico histórico para mostrar la cantidad de tráfico de red que entra a la interfaz. La figura 11.47 muestra un ejemplo de un gráfico reducido que rastrea el objeto `ifInOctects.4`.

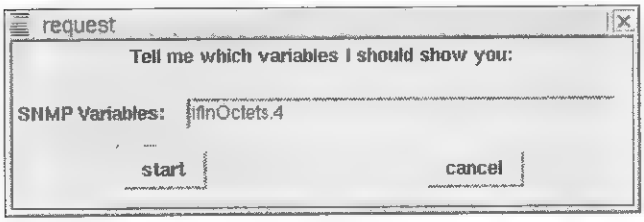


Figura 11.46 Ventana de diálogo de seguimiento.

Figura 11.47 Gráfico reducido del objeto `ifInOctects.4`.

Interface Load

Esta opción suministra un gráfico reducido para cada interfaz en el sistema objetivo. Es útil para visualizar todas las interfaces de un conjunto particular de dispositivos al mismo tiempo.

TCP Service User

Esta opción suministra un conteo de los usuarios que tienen acceso a un servicio TCP particular o que lo usan. Es una buena forma de capturar el uso intensivo de ciertos protocolos y servicios, como el acceso a la Web, transferencia de archivos, y correo.

Referencia de menús

Esta sección suministra una vista general de los menús genéricos de aplicación. Para realizar acciones específicas o tareas, se deben

revisar las siguientes secciones, a fin de obtener una explicación detallada de la funcionalidad de la aplicación.

Los menús suministrados por **tkined** se usan para tener acceso a gran parte de la funcionalidad de la aplicación. Con la excepción del menú `Tools`, todos los otros menús se describen en la sección. El menú `Tools` se explica en la última sección, "Administración de la red".

La ventana principal contiene los siguientes submenús: `File`, `Select`, `Edit`, `Structure`, `Icon`, `Options`, `Tools` y `Help`. A estos submenús se accede desde la ventana principal o desde ventanas separadas. La última característica, conocida como un menú desprendido, suministra una opción adicional para el usuario, al transformar el menú en una ventana independiente. Para desprender un menú, se selecciona el primer separador de línea de trazos localizado en la parte superior del marco del menú. Una vez hecho esto, aparecerá una nueva ventana que contiene el menú asociado. La figura 11.48 muestra la barra de desprendimiento en la parte superior del menú `Help`.

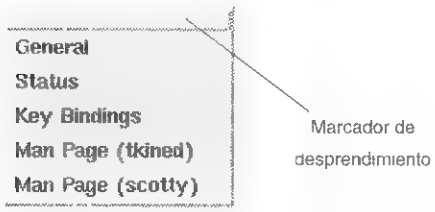


Figura 11.48 Desprendimiento del menú `Help`.

El menú `Help` suministra información genérica en línea respecto a la aplicación y características **tkined.** Uno de los elementos más importantes es la página del manual **tkined,** la cual se presentará cuando el usuario seleccione la opción `Man Page` (**tkined**) del menú. También esta incluida la página del manual Scotty, la cual suministra una vista detallada de las interfaces de programación de aplicaciones (API) para el lenguaje de herramientas de comando (TCL).

En general, se puede ejecutar la mayoría de las funciones del menú a través de un equivalente en el teclado, en lugar de seleccionar con el ratón las opciones del menú.

El menú `File` contiene opciones que se usan para crear y mantener vistas de red. La figura 11.49 muestra el menú `File`. La opción `Clear` borra la vista que presenta actualmente. La

aplicación solicitará una confirmación antes de que la operación se complete. Borrar la vista existente de manera automática crea otra vista vacía. Si el usuario borra una vista que todavía no está guardada en un archivo, la vista se perderá para siempre. La opción `Open...` del menú restaura y presenta una vista preexistente en un archivo UNIX. Presentar una vista existente creará una ventana nueva **tkined**, y el nombre del archivo de la vista se colocará en el título de la ventana.

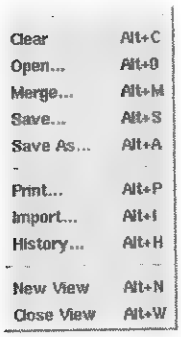


Figura 11.49 Menú `File`.

El elemento `Merge...` se puede usar para combinar diferentes vistas en una sola. La opción `Save...` guarda la vista presentada actualmente en un archivo. Al usuario se le solicita que introduzca el nombre del archivo. La opción `Save...` hace una copia de la vista y la coloca en un archivo diferente. La opción `Print...` imprime la vista activa actual.

El menú `Select` ofrece una variedad de opciones que seleccionan y destacan varios objetos. La figura 11.50 muestra este menú. La primera opción, `Select All`, selecciona todos los objetos contenidos en la vista de topología. La opción `Select Neighbours` selecciona un objeto adyacente que interconecta un objeto existente seleccionado. La opción `Select Member` selecciona todos los miembros de una vista. Las restantes opciones del menú proporcionan la capacidad de seleccionar objetos de acuerdo con el tipo, el nombre, la dirección, la etiqueta y el atributo. Cada una de las opciones presenta una ventana de diálogo, de modo que la entrada apropiada la puede recolectar el usuario. Por ejemplo, al seleccionar la opción `Type...` se presenta la ventana de diálogo que se muestra en la figura 11.51, la cual permite al usuario escoger el tipo del objeto de una lista de elementos de menú predefinidos.

Figura 11.50 Menú `Select`.

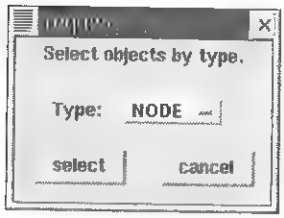


Figura 11.51 Diálogo de `Selection by type`.

La figura 11.52 muestra los tipos disponibles que se pueden escoger de la lista de selección. La lista contiene todos los tipos de objetos soportados por el software. Al escoger un tipo de objeto y dar un clic en el botón `Select` se seleccionarán todos los objetos del panel principal de topología que satisfacen los criterios. En este caso, sólo se resaltarán los tipos nodo. Las opciones `Name...`, `Address...`, y `Label...` permiten al usuario introducir una expresión de cadena que se compara con todos los objetos en la vista de topología, por ese atributo en particular. La opción restante, `Attribute...`, suministra al usuario la capacidad de definir además un criterio de selección usando definiciones de objeto adicionales. Un atributo posible, por ejemplo, es el mapa de topología al cual pertenece un objeto.

El menú `Edit` contiene varios elementos que se usan para modificar atributos de objetos y controlar la ubicación de los objetos en la vista, y opciones genéricas de edición como `Cut`, `Copy` y `Paste`. La figura 11.53 muestra el menú `Edit`.

El primer artículo, `Attribute`, es un submenú que puede adicionar, modificar o borrar un atributo de objeto. Este submenú contiene tres opciones: `Create...`, `Delete...` y `Edit...`

Figura 11.52 Selección de tipo.

Figura 11.53 Menú `Edit`.

Cada objeto definido en la vista de topología contiene propiedades (atributos) estándar y de personalización que suministran información adicional o parámetros de configuración.

Los atributos estándar asociados con todos los objetos nodo incluyen `Name`, `Address` y `SNMP: Config`. `Name` representa el nombre de *host* oficial del objeto, y `Address` es la dirección IP del objeto. La propiedad `SNMP: Config` contiene las opciones de línea de comandos que se usan cuando se ejecutan operaciones específicas SNMP en el objeto. Se pueden definir atributos adicionales del usuario. Al usar este menú, sólo se pueden borrar o editar las propiedades de personalización y los objetos `SNMP:Config`.

Para crear una nueva propiedad, primero se selecciona el objeto o el grupo de objetos con los cuales se quiere asociar esta nueva propiedad. Luego se selecciona la opción `Create` en el menú `Attribute`. Esto hará aparecer una ventana de diálogo, como se muestra en la figura 11.54. Luego se introduce el nombre del nuevo atributo en el campo `Name`. El nombre puede ser cualquiera, pero deberá ser representativo del valor que se introdujo. Por ejemplo, un nuevo atributo llamado `snmp_version`

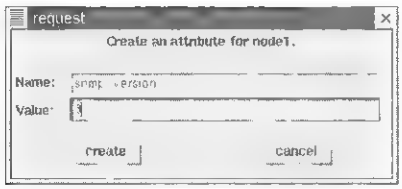


Figura 11.54 Ventana de creación de un atributo nuevo.

puede contener la versión de los protocolos SNMP soportada por el dispositivo. Si los dispositivos soportan más de una versión, el valor indica el nivel de soporte más alto. A continuación, se introduce el valor asociado con esta propiedad y, para este ejemplo, se supone que los dispositivos soportan todas las versiones. Como resultado, se introduce el valor 3, como se muestra en la figura 11.54. Finalmente, se da clic sobre el botón `Create` para guardar el nuevo atributo, y se aplica a los objetos previamente seleccionados.

La siguiente opción del menú, `Set Label...`, altera la etiqueta del objeto. La etiqueta se lista directamente abajo del icono del objeto.

El menú `Structure`, que se muestra en la figura 11.55, contiene opciones que se usan para controlar la organización del grupo y de los miembros del grupo. El menú también suministra funciones para controlar la colocación de objetos en relación con otros objetos.

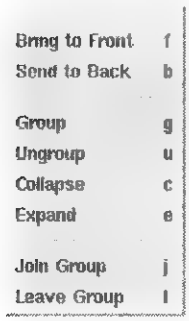


Figura 11.55 Menú `Structure`.

Las dos primeras opciones controlan cómo se presentan los objetos cuando uno de ellos se coloca sobre el otro. La primera opción, `Bring to Front`, trae un objeto oculto al primer plano, mientras que la opción `Send to Back` hace lo opuesto. Estas opciones

son más útiles cuando un objeto más pequeño creado después de crear uno más grande, se debe colocar en frente del más grande. Las opciones restantes mantienen grupos.

Las opciones del menú `Icon` asignan iconos a los elementos de objeto. Una colección grande de iconos está disponible con **tkined**, e incluye iconos para los sistemas UNIX, dispositivos de red, como enrutadores y conmutadores, e iconos de topología de red para Ethernet y otras topologías de red. Hay algunas reglas básicas acerca de los tipos de dispositivo que se pueden asignar a ciertos iconos. Por ejemplo, no se puede asignar un elemento de red (tal como FDDI) a un icono de dispositivo que representa un sistema Linux.

El uso de este menú, que se muestra en la figura 11.56, debería ser bastante intuitivo. Como se puede observar, una serie de submenús se usan para tener acceso a iconos individuales clasificados por tipo. El menú `Options` controla elementos básicos de las vistas de topología, como tamaño de página y orientación. La paleta de herramientas también se puede colocar en la parte superior de la ventana en el lado izquierdo. Los valores predeterminados de las opciones deberían ser accesibles a la mayoría de los usuarios. La figura 11.57 muestra el menú `Options`.

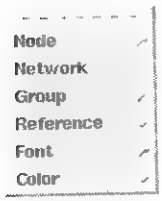


Figura 11.56 Menú `Icon`.

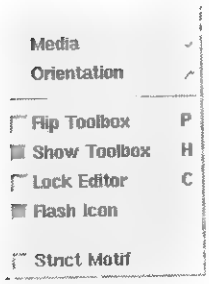


Figura 11.57 Menú `Options`.

Un vistazo a las herramientas

La tabla A.1 proporciona un resumen rápido de las herramientas descritas en este libro. Cada herramienta está categorizada en una o más clasificaciones de la administración de red relacionadas con las funciones que desempeña.

Tabla A.1 Lista de herramientas descritas en este libro.

Herramienta	Descripción	Categoría			Página
		Administración del rendimiento	Administración de configuración	Administración del diagnóstico	
arp	Hace seguimiento y controla el caché ARP		✔	✔	110
arpwatch	Hace seguimiento a la actividad ARP			✔	190
control-panel (panel de control)	Herramienta de configuración basada en GUI		✔		546
ethereal	Captura de paquetes basada en GUI			✔	194
fping	ping sofisticado	✔		✔	216
ifconfig	Configuración de interfaz		✔		119
ping	Determina el número de dispositivo(s) activos en la red	✔		✔	147
mibiisa	Agente SNMP MIB-II Sun	✔	✔	✔	396
mrtg	Herramienta graficadora del rendimiento de la red	✔		✔	474
netstat	Estadísticas de la red	✔		✔	129

Tabla A.1 Lista de herramientas descritas en este libro. (*continuación*)

Herramienta	Descripción	Categoría			Página
		Administración del rendimiento	Administración de configuración	Administración del diagnóstico	
ntop	Sigue el tráfico de la red	✔		✔	510
nmap	Explora puertos de la red			✔	224
snmpbulkwalk	Recuperación SNMP mejorada			✔	459
snmpconf	Configuraciones automatizadas SNMP		✔		462
snmpd	Agente SNMP UCD			✔	455
snmpdelta	Da formato a los datos SNMP			✔	429
snmpget	Recuperación regular SNMP			✔	431
snmpgetnext	Recupera múltiples objetos SNMP			✔	433
snmpnetstat	Recupera estadísticas de red de una entidad SNMP			✔	435
snmpset	Altera la información de configuración SNMP		✔		440
snmpstatus	Recupera datos importantes SNMP			✔	437
snmptable	Recupera la tabla de datos SNMP			✔	447

Tabla A.1 Lista de herramientas descritas en este libro. (*continuación*)

Herramienta	Descripción	Categoría			Página
		Administración del rendimiento	Administración de configuración	Administración del diagnóstico	
snmptest	Prueba la conectividad SNMP			✔	450
snmptrap	Envía trampas SNMP			✔	452
snmptrapd	Recibe las trampas SNMP de la red			✔	453
snmpwalk	Recupera un grupo o un árbol MIB completo			✔	457
tcpdump	Programa de captura de paquetes			✔	158
tkined	Editor de red que se emplea para crear mapas de topología y administrar dispositivos de red	✔		✔	582
tkmib	Explorador MIB SNMP		✔	✔	466
traceroute	Presenta caminos de enrutado	✔		✔	177
xtraceroute	Presenta información de caminos de enrutado	✔		✔	246

B

Procedimientos de instalación y notas sobre el software

Los siguientes procedimientos de instalación se utilizan para instalar las herramientas descritas en este libro. Muchos de los paquetes están disponibles en el CD-ROM que se entrega con este libro. El software también se puede obtener a través de Internet. Debido a que algunas de las herramientas pueden haber sido actualizadas desde la publicación del presente libro, se suministran indicadores de fuentes alternativas (donde sea aplicable). Adicionalmente, algunas de las herramientas están también disponibles en forma de código fuente; por ello, también se incluyen los procedimientos para construir estos paquetes.

La tabla B.1 contiene una lista de todas las herramientas analizadas en este libro. Incluye columnas que indican si la herramienta se encuentra en el CD-ROM, el directorio donde se puede encontrar y cualquier fuente alterna de Internet donde la herramienta se puede localizar. Esta información es suministrada porque algunos propietarios de herramientas las actualizan de forma periódica, y versiones más nuevas se pueden descargar directamente de la Internet.

Tabla B.1 Directorio de información de las herramientas.

Herramienta[a]	En CD-ROM	Directorio en CD-ROM	Fuente alterna
Chesapeake's Subnet Calculator	Sí	**ccci**	`http://www.ccci.com/tools/subcalc/index.html`
arp	No	ND	ND
control-panel	No	ND	ND
ethereal	Sí	**ethereal**	`http://ethereal.zing.org`

fping	Sí	**fping**	`ftp://ftp.staford.edu/pub/packages/fping/fping2.2b1.tar.gz`
			`ftp://sesame.hensa.ac.uk/ftp/mirrors/ftp.kernel.org/pub/software/admin/mon/fping2.2b1.tar.gz`
			`ftp://www.wooten.net/ftp/pub/unix/att/tcputils/fping.tar.z`

ifconfig	No	ND	ND
ping	No	ND	ND
netstat	No	ND	ND
nmap	Sí	**nmap**	`http://www.insecure.org/nmap`
ntop	Sí	**ntop**	`ftp://ftp.unipi.it/pub/local/ntop`

ND = No disponible

Tabla B.1 Directorio de información de las herramientas. (*continuación*)

Herramienta	En CD-ROM	Directorio en CD-ROM	Fuente alterna
mibiisa	Sí	**sunagent**	`http://www.sun.com/solstice/products_ent.agents`
mrtg	Sí	**mrtg**	`http://eestaff.ethz.ch/~oetiker/webtools/mrtg/pub/`
snmpd	Sí	**ucd**	Parte del paquete UCD
snmpdelta	Sí	**ucd**	Parte del paquete UCD
snmpget	Sí	**ucd**	Parte del paquete UCD
snmpgetnext	Sí	**ucd**	Parte del paquete UCD
snmpnetstat	Sí	**ucd**	Parte del paquete UCD
snmpset	Sí	**ucd**	Parte del paquete UCD
snmpstatus	Sí	**ucd**	Parte del paquete UCD
snmptable	Sí	**ucd**	Parte del paquete UCD
snmptest	Sí	**ucd**	Parte del paquete UCD
snmptranslate	Sí	**ucd**	Parte del paquete UCD
snmptrap	Sí	**ucd**	Parte del paquete UCD
snmptrapd	Sí	**ucd**	Parte del paquete UCD
snmpwalk	Sí	**ucd**	Parte del paquete UCD
snmpbulkwalk	Sí	**ucd**	`ftp://ftp.ece.ucdavis.edu/pub/snmp/ucd-snmp.tar.gz` `ftp://sunsite.cnlab-switch.ch/mirror/ucd-snmp/ucd-snmp.tar.gz` `ftp://ftp.win.or.jp/pub/network/snmp/ucd-snmp/ucd-snmp.tar.gz`

Tabla B.1 Directorio de información de las herramientas. (*continuación*)

Herramienta	En CD-ROM	Directorio en CD-ROM	Fuente alterna
snmpconf	Sí	**snmpconf**	`http://www.net.cmu.edu/projects/snmp` `ftp://ftp.net.cmu.edu/pub/snmp/snmpconf/snmpconf-V1.1.tar.gz` `ftp://ftp.net.cmu.edu/pub/snmp/cmu-snmp-V1.13.tar.gz`
tkined	Sí	**scotty**	`http://wwwhome.cs.utwente.nl/~schoenw/scotty/` `ftp://ftp.ibr.cs.tu-bs.de/pub/local/tkined`
tkmib	Sí	**ucd**	Parte del paquete UCD
traceroute	No	ND	ND
xtraceroute	Sí	**xtraceroute**	`http://www.dtek.chalmers.se/~d3august/xt/index.html`

ND = No disponible

Uso de RPM para instalar los paquetes de software

Usted puede instalar los siguientes paquetes usando el instalador de administración de paquetes Red Hat (RPM) GNOME:

- Ethereal
- MRTG
- nmap
- ntop
- xtraceroute

Muchos de los paquetes que se encuentren en el CD-ROM están en el formato de paquete RPM. En consecuencia, usted puede instalar estas herramientas usando las herramientas de instalación RPM que se encuentran en Linux. Dos versiones diferentes del RPM están disponibles en Linux y cualquiera se puede

usar para instalar la herramienta de software. La primera, llamada rpm, es una interfaz de línea de comandos. La otra, la cual proporciona una interfaz GUI, se llama **gnorpm**.

El CD-ROM contiene varios paquetes RPM y el procedimiento para instalar cada uno es el mismo. Por tanto, sólo se mostrará un paquete en el procedimiento que sigue:

Montaje del CD-ROM

1. Inserte el CD-ROM suministrado con este libro. Una vez el CD-ROM esté puesto en la unidad de CD, emita el siguiente comando:

   ```
   mount -o ro /dev/cdrom /mnt/cdrom
   ```

   Este comando montará el CD-ROM bajo el directorio `/mnt`. El resto del procedimiento usará este directorio como un ejemplo.

Uso de la herramienta de línea de comandos RPM

1. Cámbiese al directorio que contiene los paquetes RPM localizados en el CD-ROM montado:

   ```
   cd /mnt/cdrom/linux
   ```

2. Invoque la herramienta rpm para instalar el paquete deseado. En este ejemplo, se instalará el paquete xtraceroute:

   ```
   rpm  -install xtraceroute-0.8.13-1.i386.rpm
   ```

   Si usted no encuentra errores, entonces el paquete está instalado exitosamente. Si el paquete ya está instalado, entonces se presentará el siguiente mensaje de error:

   ```
   error: package xtraceroute-0.8.13-1.i386.rpm is already
   installed
   ```

Uso del GUI RPM

1. Arranque la herramienta de instalación RPM como el usuario root, emitiendo el siguiente comando:

   ```
   /usr/bin/gnorpm
   ```

   Al ejecutar este comando se presenta la ventana que se muestra en la figura B.1.

Ésta es la ventana principal de instalación **gnorpm** desde la cual usted puede instalar, consultar, verificar y remover paquetes. Usted puede ejecutar también otras tareas aquí, como buscar un paquete en particular y actualizar uno o más de ellos. Juntas, estas funciones hacen de la herramienta **gnorpm** una herramienta muy importante para el administrador de la red y del sistema.

2. Use el panel de la izquierda para inspeccionar los paquetes instalados en el sistema. Usando esta vista del árbol, usted puede navegar a través de cada una de las carpetas, revelando carpetas adicionales y, finalmente, paquetes individuales. Si usted quiere remover uno o más paquetes, simplemente seleccione el(los) paquete(s) y dé clic en el botón `Uninstall`.

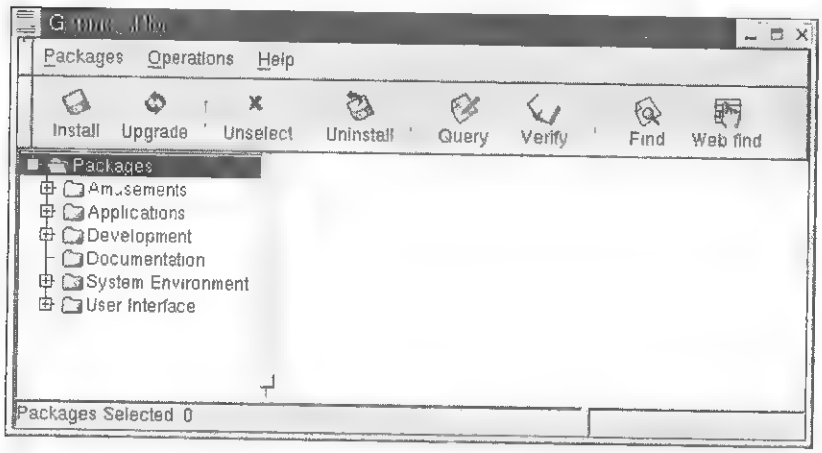


Figura B.1 Ventana principal GNOME RPM.

3. A continuación, dé clic en el botón `Install` (instalar) localizado en el extremo izquierdo de la ventana. Una nueva subventana se presentará, como se muestra en la figura B.2. Al seleccionar el submenú `Install...` del menú `Operations` aparecerá también esta ventana. La ventana presentada contiene un panel vacío, porque ningún paquete ha sido seleccionado para ser cargado en el sistema. Ahora debe seleccionar los paquetes que le gustaría instalar. Para hacerlo, debe dar clic en el botón `Add`. Entonces, aparecerá la ventana de diálogo `Add Packages` como se muestra en la figura B.3. Aquí se selecciona el paquete que le gustaría instalar.

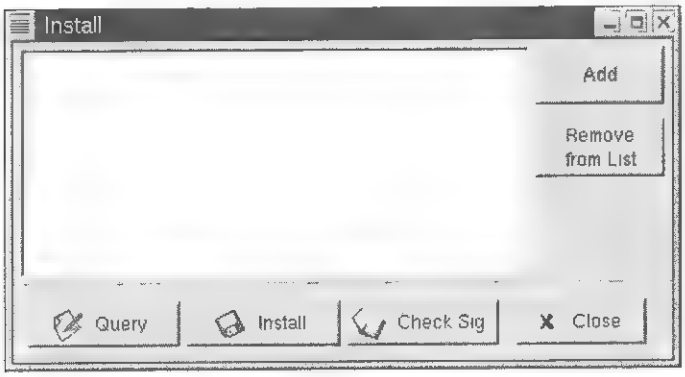


Figura B.2 Ventana de instalación.

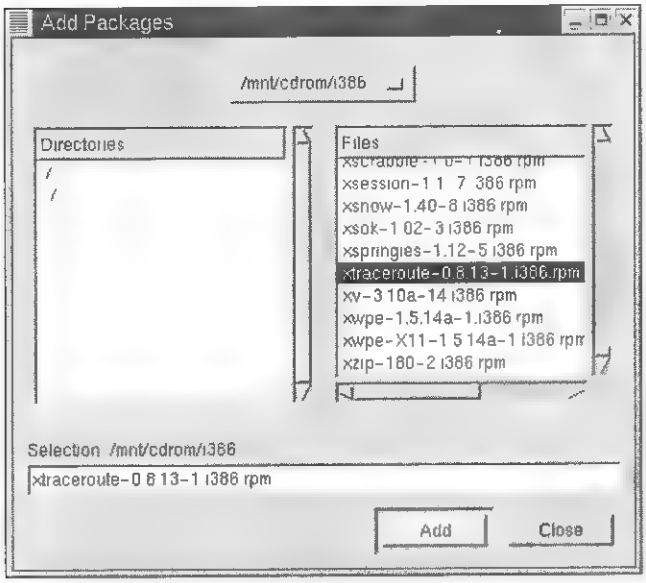


Figura B.3 Ventana `Add Packages`.

Note que la ubicación predeterminada del archivo se presenta en el directorio `/root`. Usted debe ahora navegar a la ubicación donde se encuentran los paquetes RPM. En nuestro caso, se necesita seleccionar el directorio en el cual se ha montado el CD-ROM. El CD-ROM estará montado bajo el directorio `/mnt`. Toda la ruta del paquete incluye lo siguiente:

```
/mnt/cdrom/linux
```

Una vez seleccionado el paquete `xtraceroute-0.8.13-1.i386.rpm`, dé clic sobre el botón `Add`. Esto coloca el paquete xtraceroute en la ventana `Install`, como se muestra en la figura B.4. Cierre la ventana `Add Packages` dando clic sobre el botón `Close`.

4. Antes de instalar el software, puede ser una buena idea consultar el paquete para ver qué contiene y dónde serán colocados los archivos de éste. También podrá aprender más acerca del paquete. Al dar clic sobre el botón de consulta se presentará la ventana que se muestra en la figura B.5.

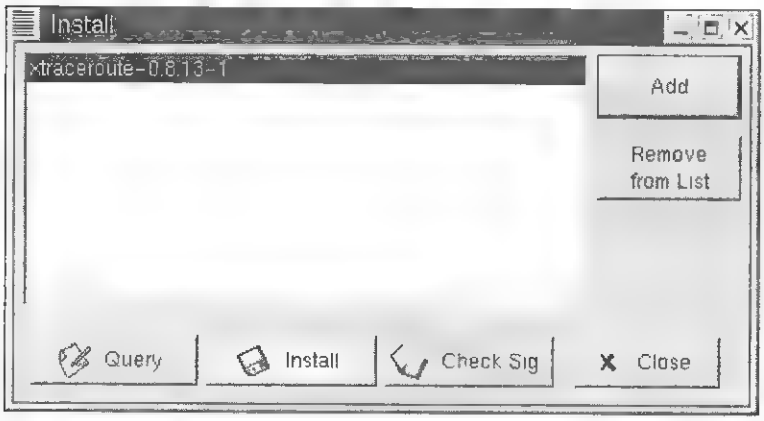


Figura B.4 Ventana de instalación con el paquete seleccionado.

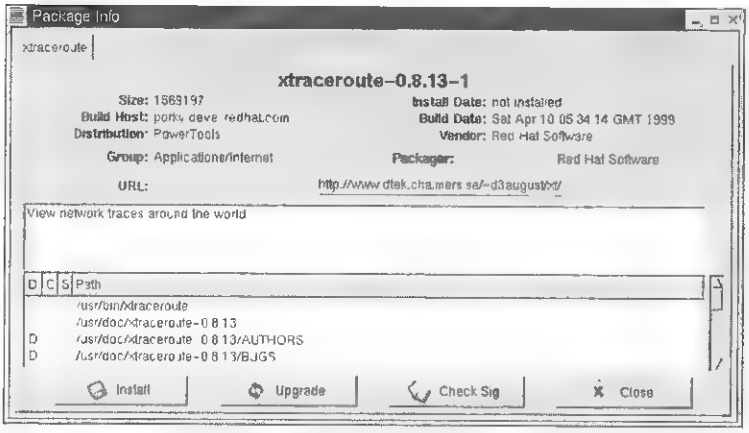


Figura B.5 Ventana de consulta del paquete.

Como se puede observar, esta ventana suministra información resumida acerca del paquete xtraceroute, una lista desplazable de los archivos relacionados con el paquete, y cuatro botones. El botón `Install` continuará la instalación, el botón `Upgrade` actualiza el paquete, si ya existe una versión anterior, y el botón `Check Sig` asegurará que el paquete contiene las firmas digitales correctas. Estas firmas se usan para verificar que el paquete está completo. Finalmente, el botón `Close` cerrará esta ventana, sin hacer nada más.

5. En este punto, usted está listo para instalar el paquete. Por tanto, dé clic sobre el botón `Install`. La ventana que se muestra en la figura B.6 presenta el progreso de la instalación usando una barra de progreso.

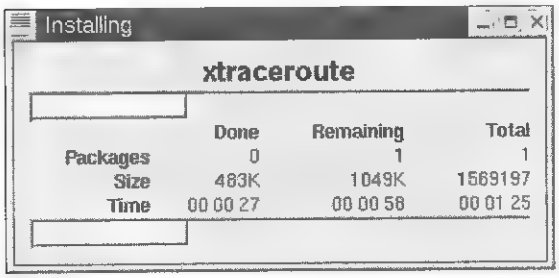


Figura B.6 Ventana de progreso.

Una vez que la ventana de progreso desaparece y no se ha presentado error, el paquete está instalado.

El instalador **gnorpm** detecta cuándo una aplicación ya está instalada. Por ello, si el usuario intenta instalar un paquete que ya está en el sistema, el software presenta un mensaje de advertencia, como se muestra en la figura B.7.

Figura B.7 Ventana de diálogo del mensaje de advertencia.

Procedimientos de instalación de paquetes que no son RPM

Los siguientes paquetes no están disponibles en formato RPM o requieren instrucciones especiales de instalación:

- calculadora de subred (*subnet calculator*)
- fping
- snmpconf
- agente Sun

Consulte al comienzo de este apéndice las instrucciones específicas para el acceso y montaje del CD-ROM.

Instalación del software Chesapeake

El procedimiento de instalación para la calculadora de subred incluye los siguientes pasos:

1. Obtener el archivo *tar* del CD-ROM o en la dirección de Internet listada en la tabla B.1. Si obtiene el archivo del sitio de Internet, debe hacer lo siguiente:
   - Llenar el formulario de registro
   - Aceptar el acuerdo de licencia
   - Descargar el archivo `subcalc.tar.Z`

   Si obtiene el paquete del CD-ROM, copie el archivo *tar* en un directorio nuevo:

   ```
   cp /mnt/cdrom/ccci/subcalc.tar.Z <directorio>
   ```

2. Desde el directorio en el cual se guardó el archivo `tar`, ejecute los siguientes comandos:

   ```
   uncompress subcalc.tar.Z
   tar xvf subcalc.tar
   [varias líneas de salida]
   ```

3. Cargue el archivo `subcalc.html` en su navegador Web favorito. Este archivo HTML está localizado en el mismo directorio de donde se obtuvo el archivo.

Instalación del software FPING

Los procedimientos de instalación para el paquete fping incluyen lo siguiente:

1. Obtener el paquete fping de uno de los sitios listados en la tabla B.1, o del CD-ROM disponible con este libro. Si está interesado en instalar sólo el binario de este programa, siga los pasos 1 y 2; no tenga en cuenta los pasos restantes.

2. Descomprima el archivo GNU:

```
gunzip fping-2.2b1.tar.gz
```

3. Desempaque el archivo en el directorio que estará usando para contener el código fuente o el binario:

```
tar xvf fping-2.2b1.tar
```

4. Normalmente, al desempacar se creará un nuevo directorio cuyo nombre empieza con **fping-2.2b1** y termina con el número de la versión del software. Cambie a este directorio:

```
cd fping-2.2b1
```

5. A continuación, ejecute el comando **configure** desde este directorio:

```
./configure

[varias líneas de salida]
```

   Después de varias páginas de salida, el programa de configuración hará algunas preguntas.

```
[preguntas]
```

6. Si no se encontró ningún error, ejecute el comando **make**:

```
make
[varias líneas de salida]
```

7. Después de que `make` ha terminado, y si no se presentó ningún error grave, instale el paquete corriendo el siguiente comando:

```
make install

[varias líneas de salida]
```

Si encuentra un problema durante la instalación, consulte el archivo `READ.ME`, el cual se puede encontrar en el directorio de nivel superior de la distribución **fping**.

Instalación del agente Sun

Para instalar el agente Sun en una versión anterior a Solaris 2.6, siga el procedimiento expuesto aquí y ejecute los comandos como el superusuario.

1. Obtenga el paquete del software de uno de los sitios listados en la tabla B.1 o del CD-ROM disponible con este libro.
2. Monte el CD-ROM en el sistema (o use el administrador de volumen).
3. Cambie el directorio al CD-ROM:

```
cd /cdrom/sol_2_6_sparc_smcc_svr/s0/Solaris_2.6/Product
```

4. Instale el paquete `SUNWascom`:

```
pkgadd -d . SUNWascom

[varias líneas de salida]
```

5. Instale el paquete `SUNWsasnm`:

```
pkgadd -d . SUNWsasnm

[varias líneas de salida]
```

6. Instale el paquete `SUNWadmi`:

```
pkgadd -d . SUNWadmi

[varias líneas de salida]
```

7. Instale el paquete `SUNWmibii`:

```
pkgadd -d . SUNWmibii

[varias líneas de salida]
```

8. Para activar el agente, reinicie el sistema o use la secuencia de comandos de inicio:

```
/etc/reboot
```

o

```
/etc/inid.d/init.snmpdx start
```

Información del paquete del agente Sun

Los agentes Sun comenzaron a ser distribuidos de manera estándar con el Solaris 2.6. El paquete binario está disponible tanto para SPARC como para Intel; por tanto, sólo se puede instalar en sistemas Solaris en cualquiera de estas plataformas. Los CD-ROM estándar Solaris 2.6 y 2.7 contienen los paquetes para los agentes empresariales Sun. Los agentes también están disponibles y se pueden descargar a través del sitio Web Sun si usted no tiene acceso al CD-ROM Sun Solaris. Los agentes se pueden instalar en las versiones 2.4 y 2.5 de Solaris. Debido a que las versiones anteriores no incluían el agente, las instalaciones de estas versiones de Solaris requerirán la utilidad del sistema pkgadd. Los paquetes requeridos y una lista de los componentes en el paquete para los agentes Sun se listan enseguida:

- SUNWascom
- SUNWsasnm
- SUNWsadmi
- SUNWmibii

SUNWascom

Este paquete contiene los archivos de configuración y comunes para los otros paquetes, los cuales se pueden encontrar en los directorios `/etc` y `/var`. Estos otros paquetes incluyen `/etc/snmp/conf/snmpd.conf` y `/var/snmp/mib/sun.mib`.

SUNWsasnm

Este paquete contiene el agente maestro `snmpdx`, el archivo MIB del agente maestro, `snmpdx.mib`, el archivo de inicio del sistema, `init.snmpdx` y bibliotecas del sistema adicionales.

SUNWsadmi

Este paquete contiene la porción DMI de los agentes. No obstante que la funcionalidad DMI no se usará, también se debe instalar.

SUNWmibii

Este paquete contiene el agente del sistema `mibiisa`.

Información de inicio del agente Sun

El agente Sun SNMP y el agente maestro están disponibles sólo en la plataforma Solaris. Éstos se incluyen de manera estándar en la versión actual de Solaris, pero se pueden instalar también en versiones previas. Para que los agentes SNMP estén disponibles todo el tiempo, deben ser arrancados cuando el sistema se inicia o reinicia. El agente Sun viene con una secuencia de comandos de inicio y es configurado cuando se instala el software. La secuencia de comandos de inicio, `init.snmpdx`, la cual se encuentra en el directorio `/etc/init.d`, arranca al agente maestro automáticamente. Una vez el agente maestro está corriendo, éste arranca cualquier subagente configurado y, de manera predeterminada, arranca el agente del sistema `mibiisa`. El listado B.1 muestra la

Listado B.1 Secuencia de comandos de inicio del agente.

```
#!/sbin/sh
# Copyright 01/17/97 Sun Microsystems, Inc. All Rights Reserved.
#pragma ident "@(#)init.snmpdx 1.11 97/01/17 Sun Microsystems"

#
# Return pid of named process in variable "pid"
#
pidproc() {
        pid=`/usr/bin/ps  e |
             /usr/bin/grep $1 |
             /usr/bin/sed -e 's/^ *//' -e 's/ .*//'`
}

#
# Kill named process(es)
#
killproc() {
        pidproc $1
        [ "$pid" != "" ] && kill -9 $pid
}

case "$1" in

'start')
      if [ -f /etc/snmp/conf/snmpdx.rsrc -a -x /usr/lib/snmp/snmpdx ]; then
           /usr/lib/snmp/snmpdx -y -c /etc/snmp/conf
      fi

      ;;
```

(*continúa*)

```
30 'stop')
31
32        killproc snmpdx
33        killproc snmpv2d
34        killproc mibiisa
35        ;;
36
37 *)
38        echo "Usage: /etc/init.d/init.snmpdx { start | stop }"
39        ;;
40 esac
41 exit 0
```

secuencia de comandos de inicio:

Al igual que la mayoría de la secuencia de comandos de inicio en Solaris, esta secuencia de comandos proporciona la capacidad de iniciar o parar al agente maestro y a los subagentes. Una instrucción suministra la lógica para determinar cuál argumento se da en la línea de comandos y adopta la acción apropiada. Si la cadena `start` se especifica, entonces se ejecuta todo entre las líneas `25` y `29`. Si se usa la cadena `stop`, entonces se ejecuta todo entre las líneas `31` y `35`. Cuando el sistema arranca, ejecuta esta secuencia de comandos con el argumento `start`. Cuando el sistema se está bajando, la secuencia de comandos se arranca con el argumento `stop`. La etiqueta `*)` (en la línea `37`) es suministrada cuando la secuencia de comandos se ejecuta en la línea de comandos. Si el argumento `start` o el argumento `stop` no son proporcionados, la secuencia de comandos presentará un mensaje indicando cómo debe usarse.

Cuando la secuencia de comandos arranca los procesos del agente, primero se asegura de que el archivo `/etc/snmp/conf/snmpdx.rsrc` existe y que es un archivo regular, y luego revisa para ver si `/usr/lib/snmp/snmpdx` es ejecutable. Si cualquiera de estas pruebas falla, la secuencia de comandos simplemente se detiene silenciosamente sin iniciar los agentes.

Cuando se invoca la secuencia de comandos para parar los procesos del agente, ésta usa la función **killproc** (líneas `8-12`) con el nombre de cada uno de los agentes, con el fin de parar los agentes. Antes de que la función **killproc** mate realmente cada proceso del agente, ésta revisa para ver si cada uno de estos procesos está corriendo, invocando la función **pidproc**. Esta función ejecuta el comando **ps** seguido de un **grep**, para coincidir con el nombre del agente. Si el agente está corriendo, retornará una identificación del proceso. Si no está corriendo, retornará la ca-

dena nula " ". El último paso es emitir el comando **kill -9** con la identificación del proceso de cada agente que está corriendo.

La secuencia de comandos está diseñada para ser usada durante y después de la iniciación del sistema. Por ejemplo, para parar los procesos del agente, use el siguiente comando:

```
# init.snmpdx stop
```

Para arrancar los agentes, use lo siguiente:

```
# init.snmpdx start
```

Instalación del paquete UCD

El proceso de instalación para el paquete UCD incluye los siguientes pasos:

1. Obtener el paquete `fping` en una de las direcciones de Internet listadas en la tabla B.1, o del CD-ROM disponible con este libro.
2. Descomprimir el archivo GNU:

   ```
   gunzip ucd-snmp-4.0.1tar.gz
   ```

3. Desempacar el archivo en el directorio que usted usará:

   ```
   tar xvf ucd-snmp-4.0.1tar
   ```

4. Normalmente, al desempacar se creará un nuevo directorio, cuyo nombre comienza con `ucd-snmp` y termina con el número de la versión del software. Cámbiese a este directorio:

   ```
   cd ucd-snmp-4.0.1
   ```

5. A continuación, ejecute el comando **configure** desde este directorio:

   ```
   ./configure

   [varias líneas de salida]
   ```

   Después de varias páginas de salida, el programa de configuración hará algunas preguntas.

   ```
   [preguntas]
   ```

6. Si no se encontró ningún error, ejecute el comando **make**:

```
make

[varias líneas de salida]
```

7. Después de que **make** ha terminado, y si no se presentó ningún error grave, instale el paquete, corriendo lo siguiente:

```
make install

[varias líneas de salida]
```

Si encuentra un problema durante la instalación, consulte el archivo de las preguntas hechas frecuentemente (FAQ) y la documentación adicional que viene con el paquete. El archivo `FAQ.txt` está localizado en el nivel superior del directorio de instalación.

El paquete UCD es de libre distribución; como tal, debe obtenerse e instalarse. Éste se suministra en la forma fuente o en la binaria. La forma binaria es más fácil de instalar, porque el paquete no necesita compilarse. Sin embargo, si quiere construir el UCD a partir de cero, necesitará un compilador y un poco de paciencia. Los procedimientos para instalar las versiones binarias y versiones fuente se suministran enseguida:

Diseño del directorio

El paquete UCD (versión 3.5) consta de los siguientes directorios:

- `agent`
- `apps`
- `etc`
- `local`
- `m`
- `man`
- `mibs`
- `ov`
- `s`
- `snmplib`

Los directorios `agent`, `apps`, `local`, `m` y `s` contienen código fuente, secuencias de comandos shell y archivos de cabacera, lo cual constituye el grueso del paquete. En particular, la ubicación `apps` contiene las fuentes para todas las herramientas SNMP (como `snmpget` y `snmpset`), y en la ubicación `agent` es donde residen el agente SNMP y los archivos fuente del manipulador de trampas (`snmpd` y `trapd`). El directorio `snmplib` contiene las fuentes para los API SNMP. El directorio `mibs` contiene una lista

de archivos MIB que soporta el agente, y el directorio `ov` contiene archivos que se usan para la integración en el paquete de software HP OpenView. El directorio `etc` contiene los archivos de configuración del agente necesarios para implementar la seguridad SNMP.

El agente UCD también se debe configurar para arrancar cuando el sistema sea subido. La versión del paquete UCD descrita en este libro no contiene una secuencia de comandos de inicio y, como resultado, se alteró la secuencia de comandos de inicio del agente Sun para incluir al agente UCD. El listado B.2

Listado B.2 Arranque del agente UCD.

```
#
# Start the LCD Agent
#
if [ -f /usr/local/share/snmp/snmpd.conf -a -x /usr/local/
sbin/snmpd ]; then
        /usr/local/sbin/snmpd
fi
```

muestra el fragmento codificado que se emplea para arrancar al agente UCD; éste se puede incluir en su propia secuencia de comandos de inicio:

De manera predeterminada, el agente UCD escucha en el bien conocido puerto `161`. Si el agente debe usar otro puerto, cambie el comando precedente, incluyendo la opción `-p` seguida por el nuevo número de puerto.

Instalación de los paquetes de software `snmpconf` y CMU

El procedimiento de instalación es el siguiente:

1. Obtener los paquetes `snmpconf` y biblioteca CMU en una de las direcciones de Internet listadas en la tabla B.1, o del CD-ROM disponible con este libro.

2. Determinar la ubicación para el software. Entre los posibles directorios están los siguientes:

   `/usr/local/snmp` y `/usr/local/snmpconf`

   Crear los directorios apropiados:

   `mkdir /usr/local/snmp` y `mkdir /usr/local/snmpconf`

3. Descomprimir el archivo CMU:

```
gunzip cmu-snmp-V1.12.tar.gz
```

4. Descomprimir el archivo `snmpconf`:

```
gunzip snmpconf V1_1_tar.gz
```

5. Desempacar el archivo CMU en el directorio que usted use:

```
cd /usr/local/snmp

tar xvf cmu-snmp-V1.12.tar
```

   a. En Solaris, es necesario modificar las fuentes.

   b. Agregue el archivo include `<sys/sockio.h>` en el archivo `snmp_extra.c` después de las otras entradas.

   Remueva los espacios adicionales cerca de la línea `versión.o` en **makefile**.

6. A continuación, ejecute el comando **configure** de este directorio:

```
./configure

[varias líneas de salida]
```

   Después de varias páginas de salida, el programa de configuración hará algunas preguntas.

```
[preguntas]
```

7. Si no se encontró ningún error, ejecute el comando **make**:

```
make

[varias líneas de salida]
```

8. Si no se encontró ningún error, ejecute el comando **make install**:

```
Make install

[varias líneas de salida]
```

9. Desempaque el archivo `snmpconf` en el directorio que usará:

```
cd /usr/local/snmpconf

tar xvf snmpconf-V1_1_tar
```

10. A continuación, ejecute el comando **configure** de este directorio:

```
./configure

[varias líneas de salida]
```

11. Si no se encontró ningún error, ejecute el comando **make**:

```
make

[varias líneas de salida]
```

12. Después de que **make** ha finalizado sin ningún error grave, instale el paquete, ejecutando lo siguiente:

```
make install

[varias líneas de salida]
```

13. Invoque un `csh` y añada la ruta de biblioteca `/usr/local/lib` a la variable `LD_LIBRARY_PATH`:

```
csh

setenv LD_LIBRARY_PATH /usr/local/lib
```

Si encuentra un problema durante la instalación, consulte los archivos `README` e `INSTALL` que vienen con el paquete. Éstos están localizados en el nivel superior del directorio en el cual fue instalado el software.

Glosario para el trabajo en red/expresiones o términos sobre la administración de red

Como muchas tecnologías e industrias, la administración de red y del trabajo en red tiene su subcultura y lenguaje propio. Este glosario se suministra para llenar el vacío y explicar algunas de las expresiones más comunes que se usan en este libro y en la industria.

Tabla C.1 Expresiones y significados.

Expresión	Significado
100Base-TX	Conocido como Ethernet rápida, es una tecnología LAN de 100 Mbps basada en el método de acceso a la red CSMA/CD usando dos pares de cable UTP de categoría 5. Tipos de medios adicionales incluyen 100Base-FX (cable de fibra óptica) y 100Base-T4 (4 pares de cable UTP categoría 3)
10Base-T	Una tecnología LAN de 10 Mbps que usa el método de acceso a la red CSMA/CD basado en especificaciones 802.3 IEEE. Tipos de medios adicionales incluyen 10base2 (Thinnet, usando cable coaxial delgado), 10Base5 (ThickNet, usando cable coaxial grande) y 10BaseFL(usando cableado de fibra óptica)
ACK	Reconocimiento; una respuesta a una pregunta específica. Por ejemplo, TCP suministra reconocimiento para cada paquete transmitido
Agente	Componente de software que se encuentra en los dispositivos de trabajo en red o en procesos del sistema operativo que responde a consultas de uno o más administradores de red
arp	Comando UNIX que presenta o manipula el caché ARP
ARP	Protocolo de resolución de direcciones: un protocolo de red es responsable de transformar direcciones de red de capa inferior (como Ethernet) a direcciones de capa superior (como el protocolo de Internet)
caché ARP	Lista de entradas ARP que consta de una dirección IP y de una dirección de enlace de datos usada por un nodo en una red TCP/IP

Tabla C.1 Expresiones y significados. (*continuación*)

Expresión	Significado
ATM	Modo de transferencia asincrónica; un estándar para el soporte de celda que se puede usar para llevar voz, datos y video en pequeñas celdas de 53 bytes de tamaño
Interfaz de unidad adjunta	Conocida como AUI; define la especificación entre una unidad de acceso a medios Ethernet (MAU) y el dispositivo DTE; especifica cómo un dispositivo Ethernet se conecta a un transmisor-receptor usado en una red 10Base5
Red vertebral	LAN o WAN central que usa una o más tecnologías de trabajo en red de alta velocidad, con redes más pequeñas que se bifurcan de ellas. Las redes vertebrales LAN conectan diferentes departamentos o grupos de usuarios. Las redes vertebrales WAN suministran a menudo conexiones a ubicaciones regionales o remotas
BER	Normas básicas de codificación. Método estándar para codificar valores de cada tipo ASN.1 como una serie de octetos de cadena o bytes
Ancho de banda	Comúnmente usada para describir la capacidad de carga de una facilidad digital de red, que se mide típicamente en bits por segundo (bps). Por ejemplo, se dice que la Ethernet 100Base-T tiene la capacidad de 100 Mbps
BootP (protocolo de arranque)	Protocolo de arranque basado en el protocolo de capa de red UDP; suministra un medio para asignar a un *host* una dirección IP dinámica
Dirección de difusión	Dirección especial que se usa para difundir información simultáneamente a todas las estaciones en una red. Los ejemplos incluyen difusión IP (134.111.10.255) y difusión Ethernet (FF:FF:FF:FF:FF:FF)

Tabla C.1 Expresiones y significados. (*continuación*)

Expresión	Significado
Dominio de difusión	Colección de dispositivos de red que comparten la misma red física o lógica. Los dominios de difusión pueden estar confinados a una serie de puertos del conmutador. Cuando ocurre una difusión en esta red, solamente la recibirán aquellos dispositivos en el dominio
Cadena de comunidad/ Nombre de comunidad	Cadena que suministra contraseñas de protección con agentes SNMPv1. Si el ministrador SNMP tiene la cadena de comunidad, le es permitido modificar cualquier objeto que se puede escribir en la MIB del agente
CMIP	Protocolo común de administración de la información; estándar ISO para administración de red que suministra algunas de las mismas funciones del SNMP
Protocolo de enlace de datos	Protocolo de red de bajo nivel que funciona capa en la 2 del modelo OSI. Hace referencia al formato del marco y al protocolo de comunicaciones. Los protocolos populares de enlace de datos incluyen Ethernet, FDDI y ATM
MIB Empresarial	Una MIB específica del vendedor definida bajo la rama empresarial y localizada bajo la rama iso.dod.internet.private. Muchos vendedores de MIB, como Cisco y 3Com, usan esta rama
Agente extensible	Agente SNMP que se puede extender para incluir funcionalidad adicional, alterando uno o más archivos de configuración o agregando algunas funciones personalizadas
Get	Operación que se usa para obtener el valor de un objeto MIB desde un agente SNMP. Sirve como la forma principal en que un administrador SNMP recupera valores MIB

Tabla C.1 Expresiones y significados. (*continuación*)

Expresión	Significado
GetNext	Similar al Get SNMP, esta operación SNMP recupera el objeto "siguiente" en el árbol MIB
MIB de recursos del *host*	Es la MIB que suministra acceso a objetos del sistema, como uso de la memoria, uso del disco, procesos que corren, etc. El agente UCD suministra acceso a estos objetos MIB
HPOV	Hewlett Packard OpenView; uno de los paquetes más populares de administración de red
Protocolo de Internet	El protocolo de Internet (IP) suministra entrega de paquete del proceso más eficaz y enrutado de la capa de servicios. Este protocolo opera en la capa 3 del modelo OSI
Agente maestro	Agente central de control que intercepta solicitudes SNMP en el puerto estándar y transmite estas solicitudes a uno o más subagentes. Usualmente, los subagentes se deben configurar para funcionar con el agente maestro
MIB	Base de información de administración; define objetos contenidos en los agentes de dispositivo SNMP usados para administrar el dispositivo
MIB-II	MIB estándar que suministra acceso a un conjunto central de objetos definidos en varios grupos. Éstos incluyen interfaces del sistema, `at`, `ip`, `icmp`, `tcp`, `udp`, `egp` y `transmission`
Explorador MIB	Programa de usuario que proporciona una vista jerárquica de objetos MIB para presentar o modificar los valores de objetos individuales

Tabla C.1 Expresiones y significados. (*continuación*)

Expresión	Significado
Multidifusión	Dirección que usa el rango de clase D (`224.0.0.0-240.0.0.0`) para la dirección IP de destino. La multidifusión representa un mecanismo más efectivo que la entrega tradicional de unidifusión para enviar información a un conjunto grande de dispositivos
OSI	Modelo estándar de interconexión abierta. El modelo define siete capas responsables de establecer la conectividad entre los dispositivos de red
OUI	Identificador único de la organización; parte de la dirección de enlace de datos que representa al vendedor del equipo; se usa para identificar marcos de enlace de datos
Administración fuera de banda	Se refiere a la habilidad de un dispositivo para soportar las capacidades de la administración remota, utilizando una interfaz especial que no se usa para la conectividad general
Paquete	Término genérico que describe los mensajes transmitidos a través de la red. Los paquetes contienen una fuente, un destino y otra información asociada
PDU	Unidad de datos de protocolo; el paquete para los tipos de mensaje SNMP. SNMP define varios tipos PDU: `get-request`, `set-request`, `get-response` y `trap`
Sonda	Dispositivo que se conecta a una red que recolecta información de rendimiento y de paquete, para un análisis y reporte posterior. Las sondas usualmente implementan RMON o algún otro protocolo de seguimiento del rendimiento de la red

Tabla C.1 Expresiones y significados. (*continuación*)

Expresión	Significado
Sondeo	Es el acto de recuperar información específica (objetos MIB) de uno o más agentes. Se dice que un administrador sondea los agentes, en busca de información requerida
RMON	Protocolo de seguimiento remoto; RMON es responsable de recolectar información de tráfico de red y de protocolo en una red sondeada. Muchos vendedores han implementado RMON en sus productos de trabajo en red
SAN	Red de área de almacenamiento. Topología de red que integra el hardware de la administración de almacenamiento masivo (es decir, bodegas de disco) a sistemas de computación que usan una infraestructura de trabajo en red basada en la estándar
Set	Operación empleada para cambiar uno o más objetos MIB en un agente. Un conjunto de operaciones SNMP requiere autenticación de seguridad para completar esta tarea
SNMP	Protocolo simple de administración de red. SNMP es el protocolo de administración que suministra capacidad de administración a sistemas y dispositivos individuales. El SNMP define un administrador, un agente y las MIB
SNMPv1	La primera versión de SNMP. Esta versión incluía definiciones del marco de trabajo central, SMIv1 y MIB-II
SNMPv2	Ampliaciones al marco de trabajo SMPv1; incluye nuevo SMI, PDU SNMP adicionales y cambios a grupos y objetos MIB-II

Tabla C.1 Expresiones y significados. (*continuación*)

Expresión	Significado
SNMPv3	Nueva versión SNMP que suministra un marco de trabajo robusto para seguridad
SMI	Estructura de administración de información; marco de trabajo general para la definición y construcción de una MIB
Máquina de estados	Modelo que contiene un número fijo de estados y acciones para la transición de un estado a otro. TCP usa un modelo de máquina de estados para las operaciones de protocolo
Subagente	Agente SNMP configurado para funcionar con un agente maestro. Todas las solicitudes SNMP vienen del agente maestro, y el subagente escucha las solicitudes SNMP en un puerto diferente
TCP	Protocolo de control de transmisión; es el protocolo de la capa 4 de transporte OSI responsable del transporte confiable de datos de una máquina a otra. TCP se usa ampliamente en la Internet
Topología	Diseño físico o lógico de una red que consta de dispositivos del trabajo en red como conmutadores, enrutadores y estaciones de trabajo individuales y servidores
Tivoli	Un paquete de administración de la red que suministra un gran número de servicios de administración. Los servicios incluyen distribución de software, administración de cuentas UNIX y administración de sistemas remotos
Trampa	Mensaje enviado desde un agente SNMP a un administrador. Los mensajes trampa pueden incluir `ColdStart`, `WarmStart`, `LinkDown` y `LinkUp`. Las trampas empresariales (`EnterpriseSpecific`) contienen información específica del vendedor

Tabla C.1 Expresiones y significados. (*continuación*)

Expresión	Significado
trapd	Programa UNIX que escucha trampas entrantes de la red usando el puerto 162. Los mensajes de trampa se pueden registrar o borrar, o pueden invocar acciones adicionales, si se ajustan a ciertos patrones o perfiles
UDP	Protocolo de datagrama de usuario; un protocolo de capa de transporte liviano que no suministra garantía de entrega de información
Unidifusión	Marco de enlace de datos enviado de un sistema a otro. Un marco de unidifusión está dirigido a dispositivos específicos en una red
URL	Localizador universal de recursos; nombre de ruta que contiene un nombre de *host* más una ruta de directorio para recuperar una página Web particular
Lista de vínculo variable	Campo con un PDU SNMP que contiene objetos MIB específicos y valores asociados
LAN virtuales	Red virtual de área local (VLAN); es una colección lógica de dispositivos de trabajo en red (servidores, estaciones de trabajo, etc.) contenidos en un único dominio de difusión
WAN	Red de área amplia; red que se extiende sobre una gran área geográfica, o se basa en tecnologías WAN, como ATM, Frame Relay, PPP y otras
WWW	World Wide Web; conjunto de sitios Web que se extienden por todo el mundo

Índice

Símbolos

. (punto), 25
  accesar, 25

A

acceso a objetos, 37
actividad de procesos del sistema, 420
acuerdo de conexión de tres vías, 78
administración de la red, 3
  mejoras y avances, 3
administrador, 4, 12
  de impresión del sistema, 548
administrador de la red
  alarmas, filtrar, 19
  aplicaciones o servicios adicionales, 18
    administración basada en las políticas, 20
    distribución de software, 19
    procesamiento avanzado de alarma, 20
    simulación de red, 20
  arquitectura
    cliente/servidor, 14
    marco de trabajo, 13
    seguimiento, 14
    soporte para plataformas, 15
  categorías de servicios, 12
  eventos, etiquetado de problemas, 19
  funciones
    aplicación, 12
    arquitectura, 12, 13
    servicios centrales, 12
  funciones del, 12
  servicios centrales, 15
    administración de la configuración, 18
    alarmas, 16
    descubrimiento de la red, 18
    detección de fallas, 15
    eventos, 16
    herramientas de reporte, 16
    interfaz, 17
    notificación de alarmas, 16
    seguimiento del rendimiento, 17
    soporte de muchos dispositivos, 16
afinación del sistema, 119
agente, 5, 23
  agente maestro, 55
  arquitectura, 56
    de maestro/subagente, 57
  comportamiento
    externo, 6
    interno, 6
  de dominio público, 6
  determinación de la funcionalidad del, 7
  extensible, 8, 9
  maestro y subagente, 55
  múltiple, 7
  problema de disputa de puertos, 56
  SNMP, 6
  subagente, 55
  vista general de, 332
agente de administración de la red, tareas, 6
agente de sistema Sun, objeto unixTime, 419
agente del sistema, 8
agente maestro de Sun, 335, 382
  agente maestro
    agentes maestros múltiples, apoyar, 387
    archivo de registro, 384
    configuración del, 382
    dos modos, 387
    información de subagente, 392
    localización del archivo de recursos, cambiar la, 386
    MIB, 388
    proceso Trap, 386
    prueba del, 388
    puerto predeterminado, cambiar el, 388
    rama, 388
    recuperación de una caída, 388
    resumen de comando de ayuda, 387
  archivo de recurso, 384

agente maestro de Sun (*continuación*)
del agente, 384
archivo de registro del agente, 383
configuración de agente, depurar la, 387
niveles de depuración, 387
números de empresa, 386
objetos MIB
fuente de lista completa, 389
lista de los más importantes, 389
objetos de configuración del subagente, 389, 392
objetos globales del agente, 389
opción
-a, 386
-c, 386
-d, 387
-h, 387
-i, 386
-m, 387
-m, modo GROUP, 387
-m, modo SPLIT, 387
-o, 386
-p, 387
-r, 386
opciones de línea de comandos, 385
disponibles, mostrar las, 387
lista de, 385
palabras clave, 383, 384
command, 384
name, 383
policy, 384
port, 384
registration_file, 384
subtrees, 383
timeout, 383
type, 384
watch-dog-time, 383
recuperación después de una caída, 386
subagente
archivo de registro, 394
configuración del, 382, 383
información disponible para el agente maestro, 392
posibles estados del, 393
registrar, 382
sunCheckSubAgentName, 391
sunMasterAgentConfigurationDir, 390
sunMasterAgentMaxAgentTimeOut, 391
sunMasterAgentPollInterval, 391
sunMasterAgentResourceConfigFile, 389
sunMasterAgentStatusFile, 389
sunMasterAgentTrapPort, 390
sunSubAgentAccessControlFile, 394
sunSubAgentExecutable, 394
sunSubAgentID, 393
sunSubAgentName, 395
sunSubAgentPortNumber, 394
sunSubAgentProcessID, 395
sunSubAgentRegistrationFile, 394
sunSubAgentStatus, 393
sunSubAgentSystemUpTime, 395
sunSubAgentTimeout, 393
sunSubAgentVersionNum, 395
sunSubAgentWatchDogTime, 396
vista de sunMasterAgent, 390
vista de sunSubAgentEntry, 392
agente *proxy*, 11
como puente entre las versiones de SNMP, 11
suministrar conectividad a la red, 11
agente SNMP de Linux
archivo de configuración de agente, 339
comando kill, 341
comando ps, 341
comando snmpget, 341
palabra clave, 339
archivo ejemplo de configuración de agente, 368
asignación de puerto, agente maestro, 369
cadena maximum load (carga máxima), 342
comando
df, 345, 346
disk, 344
echo, 360
exec, 352, 353, 355
kill, 370
load, 342
pass, 355, 358
proc, 346
ps, 346, 353, 355
tail, 367
uptime, 342, 352
view, 350, 351
w, 342

comandos
adicionales, 342
externos UNIX, invocar, 352
descripción, 338
diskTable, 376
dskAvail, 379
dskDevice, 378
dskEntry, 377
dskErrorFlag, 379
dskErrorMsg, 380
dskIndex, 377
dskMinimum, 378
dskMinPercent, 378
dskPath, 377
dskPercent, 379
dskTotal, 379
dskUsed, 379
espacio del disco, seguimiento del, 344
extCommand, 375
extensión del agente, 352
extErrFix, 376
extIndex, 375
extNames, 375
extOutput, 376
extResult, 376
extTable, 374
funciones, lista de, 339
laContig, 381
laErrMessage, 382
laErrorFlag, 382
laIndex, 381
laLoad, 381
laNames, 381
loadTable, 380
MIB, soportado, 338
objeto MIB prEntry, 372
objetos
de diskTable, lista de, 377
extTable, lista de, 375
procTable, lista de, 372
UCD MIB, lista de, 371
opción
-a, 369
-C, 370
-c, 370
-d, 369
-f, 370
-L, 370
-l, 370
-p, 369
-q, 370
-V, 370
opciones de línea de comandos, 368
palabra clave proc, 372
prCount, 373
prErrFix, 374
prErrMessage, 374
prErrorFlag, 373
prIndex, 372
prMax, 373
prMin, 372
prNames, 372
procesos
del sistema, seguimiento a los, 346
que corren actualmente, limitar la cantidad de, 347
UNIX, seguimiento crítico, 368
procTable, 371, 372
puerto, 369
secuencia de comandos
estándar de arranque, 341
passtest, 359
perl para mantener el archivo de mensajes, 362
seguimiento al promedio de carga del sistema, 342
seguridad del agente, 348
tipos de objetos MIB de agente soportados, 357
UCD MIB, 371
umbral de carga del sistema, establecer el, 342
uso de la facilidad pass, 355
agente SNMP de Linux (snmpd), 334
agente SNMP de Sun, 336, 396
agentDescr, 407
archivo de configuración, 398
probar nuevos, 397
banderas de estado, ejecutar, 411
comando
netstat, 413
ps, 408
comparación entre los objetos MIB-II y Sun MIB, 403
configuración del agente Sun, 398
diferencias de MIB Sun, 402

agente SNMP de Sun (*continuación*)
ejemplo
de configuración, 400
de un archivo de configuración, 400
fallas de seguridad, abordar, 400
grupo
sunHostPerf, 413
sunHostPerf, información adicional, 413
sunHostPerf, objetos más útiles, 413
sunProcessTable, 408
hostID, 407
inhabilitar el acceso de escritura, 396
lista de control de acceso (ACL), 400
mensajes de trampas, inhabilitar los, 397
MIB
privada de Sun, 396
Sun enterprise, 406
MIB-II, 396, 402
modo depurar, 397
motd, 407
niveles de depuración, 398
número de vendedor amplio Sun enterprise, 402
objetos MIB del agente Sun, 402
objetos MIB Sun enterprise, 405
grupo de rendimiento Sun Host, 413
grupo de tabla de proceso Sun, 408
grupo sunSystem, 405
grupos de, 405
opción
-a, 397
-c, 397
-d, 397
-p, 398
-r, 396
opciones
de comandos del agente maestro, 397
de configuración, 398
de línea de comandos, 396
palabras clave, 398
kernel-file, 400
managers, 400, 402
read-community, 400
sysdescr, 399
syslocation, 399
system-group-read-community, 400
system-group-write-community, 400
trap, 399
trap-community, 400
write-community, 400
psParentProcessID, 410
psProcessCpuTime, 410
psProcessID, 409
psProcessName, 412
psProcessSize, 410
psProcessState, 411
psProcessStatus, 412
psProcessTTY, 411
psProcessUserID, 412
psProcessUserName, 412
psProcessWaitChannel, 411
puerto predeterminado, cambiar el, 398
rsIfCollisions, 416
rsIfInPackets, 415
rsVIntr, 415
rsVpagesIn, 413
rsVpagesOut, 413
rsVSwapIn, 414
rsVSwapOut, 414
seguridad, 397
servicios, lista de, 396
tabla sunProcessTable, índice para la, 408
unixTime, 408
vista
de sunHostPerf, 414
de sunProcessTable, 409
de sunSystem, 406
agente SNMP de Sun (mibiisa), paquete UCD, 420
agente snmpdx
véase agente maestro de Sun, 385
agente Sun
información
de inicio del, 652
del paquete del, 651
instalación del agente Sun, 648
secuencia de comandos de inicio del agente, 652
agente UCD, 334
archivo de configuración del, 419
arranque del, 656
comandos del, 340
instalación del paquete UCD, 654

agentes SNMP, 140, 331-416
  acciones personalizables, 333
  administración
    de memoria/disco, 333
    del proceso, 332
  agente
    maestro de Sun (snmpdx), *véase* agente maestro, 335
    SNMP de Linux (snmpd), *véase* agente SNMP de Linux, 334
    SNMP de Sun (mibiisa), instalando versiones anteriores de Sun OS, 336
    SNMP de Sun (mibiisa), *véase también* agente SNMP de Sun, 336
  arbitraje de puertos, 336
  configuración
    flexible, 332
    remota, 332
  consideraciones de seguridad, 337
  extensible, 333
  funcionalidad de los, 332
  funciones, lista de, 332
  MIB del agente, 336
  objetos de rendimiento, 333
  SNMP de Linux, ventajas de usar, 335
  soporte
    de agente maestro, 333
    de versiones, 333
    MIB extendido, 332
    MIB-II, 332
  vista general funcional de los agentes, 334
agentX, 58
alarmas, 16
análisis de tendencias, 3
analizador de protocolo, 158
árbol MIB, recorrer el, 47
ARCHIE, 72
ARP, 69, 99, 100
  bandera
    C, 113
    M, 117
    P, 113
  borrar un caché ARP, 116
  caché, 102, 103, 110
  cadena incomplete, 113
  campo
    Address, 112
    de operación, 102
    de tipo de hardware, 102
    Flags Mask, 112
    hlen, 102
    HWaddress, 112
    HWType, 112
    iface, 113
  campos
    de protocolo, 102
    del paquete, 101
    plen, 102
  carga de vínculos ARP usando un archivo, 118
  descripción, 110
  ejemplo del comando -v, 111
  emisor
    HA, 102
    IP, 102
  estrategias de acceso telefónico remoto, 118
  formato
    de carácter de comando, 111
    de direcciones de enlace de datos, 103, 104
    de la palabra clave, 111
    de mensaje, 102
    del paquete, 101
  muestra, 112
  opción
    --display, 114
    -a, 114
    -d, 116
    -f, 118, 119
    -n, 114
    -s, 117
    netmask, 118
  opciones
    de campo Flags Mask, 112
    de línea de comandos, 111
  palabra clave
    pub, 118
    temp, 118
  privilegios del superusuario, 111
  proxy ARP, 113, 118
  refrescar los vínculos caché, 102
  respuesta, 101
  secuencia de comandos de inicio, 119
  solicitud, 100

ARP (*continuación*)
target IP, 102
tipo de hardware, 102
ventajas, 100
arpwatch
descripción, 190
dirección Ethernet cambiada, 192
flip flop, 193
nueva actividad, 194
nueva estación, 191
opciones de línea de comandos, 191
tipos de mensajes, 191
arquitectura, 13
del protocolo TCP/IP, 63
ASN.1, 27, 42
tipos de aplicación, 28
tipos universales, 28
autoridad de números asignados en Internet (IANA), 40
autoridad del registro IEEE, 104

B

back-off, 139
base de información para la administración, *véase* MIB
bloque de control TCP (TCB), 308
véase también MIB-II, grupo TCP

C

caché ARP, 102
agregar una entrada de, 116
presentación del, 111
capa
de aplicación, 62
de presentación, 62
de sesión, 63
de transporte, 63
física, 63
centro de información de red (NIC), 87
check reachability, 611
códigos de respuesta SNMP, 52
colisión, 139
comando
grep, 135, 554
hostname, 551
Linuxconf, 549
netstat, 531, 568
salida del, 313
nmap, 609
route, 143, 567
snmpdx del agente maestro, 387
snmpget, 420, 475
sudo, 159
target
véase herramienta MRTG, comando target, 498
vmstat, 609
wc, 458
comandos UCD
buscar objetos MIB, acceso aleatorio para, 426
comando all, 428
descripción, 420
especificar
todos los módulos MIB, 427
versión SNMP, 422
formato rápido, *véase* opción -q, 424
información de depuración, 425
opción
-c, 425
-d, 425
-f, 423
-h, 423
-M, 427, 428
-m, 427
-q, 424
-R, 426
-S, 424
-s, 424, 429
opciones
comunes de línea de comandos, 422
de pantalla, lista de, 423
de presentación, 423
operacionales, 425
lista de, 426
salida
de objetos MIB, 425
de opción -q, 424
get-request SNMP del objeto MIB system.sysContact, 425
sintaxis básica del comando, 422
sysUpTime, 424
variable
MIBDIR, 428
MIBS, 428

PREFIX, 428
SUFFIX, 428
variables de entorno, 428
lista de, 428
comité IEEE 802.3, 314
completamente calificada, 498
componente de sistema de red, agentes *proxy*, 11
componentes de la administración de redes, 4
agentes
manipulación de MIB, 6
sistema UNIX sustituto, 6
MIB, 8
opcional, 4
agentes *proxy*, 4
requeridos, 4
administradores, 4
agentes, 4
MIB, 4
comprimir, 165
comunidades SNMP, 40
conexiones, 75
virtuales, 73
configuración de interfaces
véase ifconfig
configurador de la red
campo
Device (dispositivo), 567
Gateway (puerta de enlace), 567
Netmask (máscara de red), 567
Network (red), 567
campos de entrada estáticas de enrutado, 568
configuración de enrutado, 566
entrada de ruta estática, 567
lista de funciones, 550
panel de enrutado, 566
usos del, 549
véase también panel de control Linux, 549
congestión de tráfico, sondeo, 14
conjunto de protocolos TCP/IP, 59, 60
consumo del disco, precisión del, 502
control de configuración, 420
control de flujo
véase PPP, control de flujo, 558
cuenta de saltos, 143

D

datos en secuencia, 73
número
de secuencia, 73
de segmento, 73
demonio de impresora de línea (LPD), 70, 549
depuración, 119, 129, 148
problemas de la red, 183, 184
protocolos de red, 168
desarrolladores
agente, 6
desempeño de red, 148
detección de fallas, 15
DHCP, 69
diagrama funcional de agente, 8
difusión, 100
dirección de, 124
dominio de, 20
dirección
de enlace de datos, 110, 124
tabla, 110
transformación (*mapping*), 110
vinculación (*binding*), 110
IP, 124
MAC, 528
direcciones estándar de máscara de red IP, 126
diseño del directorio, 655
dispositivo SNMP, recopilación de información de interfaz, 435
dispositivos, 418
conexión, ver el estado de la, 531
configuración automatizada con la herramienta snmpconf, 462
descubrimiento automático de las interfaces, 483
información importante de los, 418
interfaz, validar con cfgmaker, 483
problemas de rendimiento, 474
rendimiento real, determinar el, 516
vertebrales, 474
véase también herramienta MRTG, palabra clave target
distribución de software, 19
DNS, 69

E

editor de la red, *véase* herramienta tkined, 581
ejemplos de objetos MIB, 31
enrutador lateral, 474
estadísticas del protocolo, 419
estado, 31
  de red, seguimiento, 129
estándar MIB II
  grupo interface, 33
  información relacionada con el rendimiento, 33
estructura del grupo interface MIB-II, 35
estructura MIB
  explorar dinámicamente, 434
  localizaciones lexicográficas, 433
  véase también herramienta snmpgetnext, 433
ethereal, 460, 518, 543
  captura de paquete, optimizar la, 199
  descripción, 194
  elemento de menú Follow TCP Stream, 212
  establecimiento de preferencias y opciones, 208
  guardar los datos capturados en un archivo UNIX, 207
  instalación de, 642
  opción
    Color Display..., 216
    Match Selected ..., 216
  opciones
    de presentación, 196, 211
    funcionales, 196
  Printing (imprimir), 209
  submenú Summary..., 214
  tráfico de red, captura del, 199
Ethernet
  captura de red, filtro de, 205
  filtro(s)
    de captura de paquete, 203
    de presentación, 203
    permanentes, 203
  paquete(s)
    campo Ethernet II, 201
    examinar, 201
    formato hexadecimal, 202
    porción ARP, 202
  tamaño del mensaje estándar, 124
  tipos de paquetes de captura, 200
  uso de filtros, 203
eventos, 16
explorador de puerto
  véase nmap, 224
explorador MIB, 26, 27
  véase también herramienta tkmib

F

Fast Ethernet, 314
FDDI, tamaño del mensaje estándar, 124
filtro
  de difusión IP, 205
  de presentación FTP, 206
FINGER, 69
firewall (barrera de seguridad), 67
formato
  de direcciones de enlace de datos, 103, 104
    códigos OUI de vendedor, 105
    lista de vendedores, 105
  de secuencia TCP, 81
  del paquete, 101
  hexadecimal, 126
fping
  descripción, 216
  identificación de sistemas activos, 218
  información de rendimiento, 221
  instalación de, 648
  lista de opciones
    de presentación, 217
    operacionales, 222
  opción
    -A, 220
    -a, 217, 218
    -B, 222
    -c, 224
    -e, 221
    -s, 221
    -u, 218
  opciones
    de presentación, 217
    operacionales, 222
  solicitud ICMP, 216
FTP, 60, 67, 69

autenticación del usuario, 76
interfaz de comando, 76
sesiones, 76
FTP Proxy, 228
fuerza de trabajo de ingeniería de Internet (IETF), 22
funciones del administrador, 12

G

GetBulkRequest
campos no repetidores, 52
campos repeticiones máximas, 52
GetNextRequest, 47
GetRequest, 45
campos
pdu, 46
request-id, 46
vínculos variables, 46
formato de mensaje, 44
GetResponse, 48
campos
estado de error, 48
identificación de solicitud, 48
índice de error, 48
pdu, 48
vínculos variables, 49
formato de mensaje, 48
glosario para el trabajo en red, 659
gnorpm, 643
GOPHER, 72
graficador de tráfico de multi-ruta (MRTG)
véase herramienta MRTG
grupo at, 275
grupo de interfaces
descripción, 33
dirección física, 34
estado operacional, 34
estructura, 34
ifDescr, 35, 37
ifEntry, 36, 37
ifIndex, 35, 37
ifInOctets, 35, 36, 37
ifNumber, 34, 37
ifOutQLen, 37
ifTable, 35, 37
mtu, 33
tipo, 33
último cambio, 34
velocidad, 33
grupo de multidifusión, 144
gzip, 165

H

herramienta
de cambio de tamaño, 589
de configuración, 549
herramienta MRTG, 474-510
archivo de configuración, construcción del, 479
campo
Max Speed, 478
System, 477
comando
Directory, 493
IconDir, 489
Refresh, 489
target, 498
WithPeak, 492
WriteExpires, 489
comandos
globales, 488
opcionales objetivo, 490
componentes adicionales, 506
configuración básica, 479
datos de lectura mrtg, generar, 503
descripción, 474
descubrir interfaces, 484
entrada target, 482
espacio del disco, seguimiento del, 502
formato de reporte HTML, usar el, 483
graduación de ejes, 492
gráfico(s)
anual, 479
diario, 478
semanal, 479
modificar el color, 478
modificar la presentación de los, 493
indexmaker,
argumentos de línea de comandos, 496
sintaxis de, 496
índice maestro, 496
para los dispositivos a los que se les ha hecho seguimiento, 499

herramienta MRTG (*continuación*)
información adicional de seguimiento, 498
instalación de la, 642
mensajes de error
cadena de comunidad correcta, 509
detección de archivo de bloqueo, 509
error de sintaxis, 510
falla al alcanzar el dispositivo target, 510
palabra clave target faltante, 509
palabra clave WorkDir faltante, 510
Rateup WARNING, 508
muestra de salida de información WorkDir, 487
nombres objetivo, reiniciar, 495
nombres objetivos especiales, 494
carácter $, 494
carácter ^, 494
carácter _, 494
objeto
ifInErrors, 500
ifInOctects, 484
IfOutOctects, 484
ifSpeed, 478
ifType, 478
MIB ifInOctects, 478
MIB ifOutOctects, 478
sysContact, 477
sysLocation, 477
sysName, 477
objetos predefinidos MRTG SNMP, 500
opciones
de línea de comandos de la secuencia de comandos df2mrtg, 504
de personalización, 478, 490
palabra clave
AddHead, 491
Maxbytes, 484
PageTop, 485, 490
Suppress, 492
target, 482, 483
Title, 485
Unscaled, 492
WorkDir, 486
palabra clave options, 493
conmutador absolute, 494
conmutador gauge, 494
conmutador growright, 494
conmutador noinfo, 494
conmutador nopercent, 494
conmutadores booleanos, 493
palabras clave, lista de, 483
personalización de los reportes, 488
programa para ejecutar, 488
programación con dispositivo cron, 487
recolectar información de otras fuentes, 502
reporte
de uso del disco, en porcentajes, 505
del espacio del disco, 503
mensual, 479
Web, 477
resolución de problemas MRTG, 508
secuencia de comandos df2mrtg, 504
opción -f, 504
opción -p, 506
opción -s, 504
secuencia de comandos indexmaker, 496
sintaxis
de palabras clave, 483
de WorkDir, 486
uso del MRTG, 486
usos, lista de, 475
utilidad cfgmaker, 479
ejemplo del archivo de configuración, 481
sintaxis, 479
vista
de operación básica, 476
de reportes semanales, mensuales y anuales, 480
general de una página Web, 476
herramienta netstat, descripción, 129
herramienta ntop, 510-544
actividad de paquete de multidifusión, 522
beneficios, 511
campo
Local Domain Name (nombre de dominio local), 524
Nw Interface Type, 524
Sampling Time (tiempo de muestreo), 524
desventajas, 511
filtros, 518

flujos de red, 518
instalación de la, 642
modo terminal, uso del, 543
modo Web, 512
  cómo asegurar los reportes, 541
  lista de reportes, 514
  opción -w, 512
  salida de arranque, 513
modos de vista, 512
opción
  -F, 518, 542
  -h, 544
  -i, 536
  -p, 541
opciones de línea de comandos, 541
políticas de seguridad, 511
presentación del modo terminal, 544
reportes
  campo Active Since, 533
  campo Data Received Stats, 531
  campo Dropped by ntop, 525
  campo Duration, 533
  campo IP Traffic, 527
  campo MAC Address, 528
  campo non-IP Traffic, 527
  campo Total, 527
  campo Total Traffic, 527
  categoría de protocolo IP, 538
  cómo asegurar los, 541
  estadística Broadcast Pkts Sent, 531
  estadística Total Data Sent, 531
  Global Protocol Distribution (distribución global de protocolo), 527
  lista de reportes de categoría estadística, 522
  lista de reportes de datos recibidos, 514
  lista de reportes de envío de datos, 519
  lista de reportes de tráfico IP, 533
  primer reporte de estadísticas de tráfico, 525
  reporte de distribución, 538
  reporte de estadísticas de *hosts*, 528
  reporte de estadísticas de multidifusión, 522, 523
  reporte de estadísticas globales de tráfico, 539
  reporte de flujos de red, 518
  reporte de *host*, 529
  reporte de *host* individual, 530
  reporte de matriz de tráfico, 535, 536
  reporte de rendimiento, 516, 517, 532
  reporte de rendimiento de datos enviados, 521
  reporte de subred del uso del protocolo, 540
  reporte de todos los protocolos, 515, 519, 520
  reporte de tráfico IP de remoto a local, 533
  reporte de uso local, 537
  reporte de uso local (parte dos),538
  reporte de uso local (parte uno), 537
  reporte IP, 516, 519, 520
  reporte IP remoto/local, 534
  segundo reporte de estadísticas de tráfico, 529
  tabla Active TCP Session, 531
  umbrales de rendimiento del reporte de matriz de tráfico, 535
seguridad, 541
  ventana de inicio de sesión, 542
uso del modo Web, 512
vista de página Web principal, 513
herramienta RPM
  línea de comandos, uso de la, 643
  uso del GUI, 643
herramienta snmpconf, 462
  carácter
    A, 463
    I, 463
    S, 462
  descripciones de campo, 463
  ejemplo del archivo, 463
  mensaje de error, 465
  opción
    -quiet, 465
    -version, 465
  opciones de línea de comando, 465
  salida, 464
  sintaxis, 463
  usado con el comando ping, 465
herramienta snmpdelta, 429
  opción
    -m, 430

herramienta snmpdelta (*continuación*)
-R, 429
-T, 431
opciones de línea de comandos, 430
salida de opción
-m, 430
remote-gw, 429
herramienta snmpget, 431
salida sysDescr y sysContact, 432
sintaxis del comando, 432
herramienta snmpgetnext, 433
herramienta snmpnetstat, 435
información
de enrutado, presentación de la, 438
de interfaz, presentación de, 436
opción
-I, 436, 437
-i, 437
-I, con argumento interval, 438
-P, 439
-r, 438
-s, 437, 439
opciones de línea de comandos, 436
salida de opción
-i, 437
-s, 439
salida del enrutador Cisco 7000, 437
herramienta snmpset, 440
conjunto de no retorno, 443
funciones, lista de, 441
inhabilitar interfaces, 443
sintaxis, 441
tipos de objeto, 442
herramienta snmpstatus, 444
información, lista de, 444
opción de comando foreach, 446
salida de, 446
sintaxis, 444
herramienta snmptable, 447
carácter separador de columna, cambiar el, 449
ifIndex, 449
NetAddress, 449
opción
-b, 449
-f, 448
opciones de la línea de comandos, 448
PhysAddress, 449
salida
de ejemplo de la tabla TCP, 448
de muestra atTable, 449
sintaxis, 447
tablas, lista de, 447
herramienta snmptest, 450
modos de operación, 450
herramienta snmptranslate, 450
opción -d, 451
opciones de línea de comandos, 451
salida de opción -d, 451
herramienta snmptrap, 452
configuración de trampas, mejores prácticas, 453
salida, 453
sintaxis, 452
soporte de formatos, 452
herramienta snmptrapd, 453
archivo syslog.conf estándar con entrada trap, 456
opción
-D, 454
-d, 454
-P, 454
-q, 454
-s, 456
salida de -d y -P, 454
secuencia de comandos de inicio UCD, 455
herramienta snmpwalk, 457
búsqueda predeterminada, 459
salida de información desde una búsqueda limitada, 458
sintaxis para recorrer toda la MIB, 458
versus snmpbulkwalk, 459
versus snmpget, 457
herramienta snmpwalkbulk, 458
captura de paquetes, 460
compatibilidad con la versión snmp, 462
versus tcpdump, 460
herramienta tcpdump, 207, 460, 518, 543
descripción, 158
ejemplos de, 177
herramienta tkmib, 466
capacidad para dar formato a los datos como entrada para otras aplicaciones, 466

cuadro de diálogo
de la cadena de comunidad, 471
número de puerto, 470
descripciones de los botones de la ventana principal, 466
elementos del menú, 467
menú
File, 467
Mib, 467
Options, 470
Options, submenús, 469
ventana
de diálogo de carga del archivo MIB, 469
principal, 467
vista de árbol para la rama Cisco, 470
herramienta traceroute, descripción, 177
herramientas
de reporte, 16
directorio de fuentes, 640
instalación de paquetes que no son RPM, 648
instalador de administración de paquetes Red Hat (RPM) GNOME, 642
lista de, 635
pasos de la instalación RPM, 643
ventana principal GNOME RPM, 644
herramientas snmp, 417-471
actividad de procesos del sistema, 420
control de configuración, 420
latido del sistema, 419
lista de, 418
rendimiento de la interfaz, 420
seguimiento de la ruta, 420
herramientas UCD SNMP, 421
HTP, 70

I

ICMP, 70, 93, 614
administración de congestión en la red, 96
depuración con, 93
mensaje de tiempo excedido, 98
redirigir mensaje, 95
retirar la fuente, 96
tipos de mensajes de error, 94
véase MIB-II, grupo ICMP
identificadores únicos de organización (OUI), 104
IETF, 42
ifconfig, 119, 565
bandera up, 123
campos de opciones de presentación, 122
Bcast, 122
HWaddr, 122
inet addr, 122
Link encap, 122
Mask, 122
control del estado de administración, 123
descripción, 119
desempeño de la interfaz, prueba comparativa del, 122
funciones, lista de, 119
información modificable de interfaz, lista de, 124
interfaces disponibles, listado de, 120
interfaces lógicas, 127
configuración, 127
visualización, 128
interfaz
inhabilitación, 123
número de interrupción, 122
modificación de los parámetros de la interfaz, 123
modo promiscuo, 127
opción -a, 121
opciones
de modo y estado, 123
palabras clave y, 120
palabras clave
allmulti, 127
arp, 127
down, 123, 128
media, 127
pointtopoint, 127
promisc, 127
running, 123
up, 123
parámetros para configuraciones especiales, lista de, 126
inetd, 75
información
de enrutamiento de multidifusión, 144

información (*continuación*)
de inicio de sesión TCP, 213
DNS, 551
sobre el rendimiento, 3
informRequest, 52
instalación de los paquetes de software usando RPM, 642
instalador de administración de paquetes Red Hat (RPM), 642
interconexión de sistemas abiertos (OSI) *véase* modelo OSI, 62
interfaces de red, configuración de, 119
interfaz céntrica, 496
interfaz de abstracción de flujo, 74
interfaz de programación de aplicaciones (API), 14, 628
interfaz de usuario gráfica (Graphical User Interface) (GUI), 17
interfaz gráfica de usuario Linux (GUI) *véase* panel de control Linux, 545
interfaz punto a punto (PPP) *véase* PPP, 558
interfaz virtual, 127
véase ifconfig, interfaces lógicas, 127
inundación de trampas, 453
inundar una red, 157
IP, 84
ARP, 69, 99, 100
clases de direcciones, 86
primarias, 86
direccionamiento, 86
de las subredes, 88
direcciones
de ciclo de retorno, 87
de difusión, 87
entrega no confiable, 85
ICMP, *véase* ICMP
IPv4, 85
direcciones, 86
IPv6, 85
lista de características y servicios, 85
paquetes en rutas diferentes, 86
protocolo
de resolución de direcciones (ARP), 69, 99, 100
no orientado a la conexión, 85
subredes, 88
IRC, 72
ISO 8802-3, 314

K

kernel configurator (configurador del kernel), 549

L

latido del sistema, 419
lenguaje de herramientas de comando (TCL), 628
límite de seguridad, 40
listado de directorio FTP, 214
listas de control de acceso (ACL), 20
localizaciones lexicográficas, 433
longitud de la cola del paquete de salida, 34
LPD, 70

LL

llamada a procedimientos remotos (RPC), 70, 605

M

maestro y subagente, 55
máquina
de estado TCP, 80
de estados, 79
marco, 65
máscara de subred de red *véase* netmask, 125
máxima unidad de transferencia (MTU), 124
mayores generadores de tráfico, 535
mensajes del sistema subir/bajar, 419
menú, desprendimiento del, 628
MIB, 6, 8, 25-41, 254
3cProd-mib, 10
3cWeb-mib, 10
acceso a objetos, 37
actual (current), 32
Cisco-mib, 10
de impresora, 10
descripción, 33
del sistema, 38
DNS, 10
ejemplo de estructura, 32
estado, 31
de administración, 34
estándar, 38

FDDI, 10
formato
círculos rellenos, 33
cuadrados rellenos, 33
descripción de gráficas jerárquicas, 33
grupo interfaz, 33
histórico (historic), 32
hojas, 25
identificador de objeto, 31
información
de seguimiento del rendimiento, 33
relacionada con el rendimiento, 33
ISDN, 10
lista
de propiedad, 10
de, estándar, 10
modo de acceso, 31
nombre del objeto, 31
notación punteada, 25
forma resumida, 33
identificador de objeto, 25
identificador numérico, 25
objetos, 25
obsolete, 32
organización, 25
privada, 38
rama private (4), 40
recurso *host*, 10
snmpdx.mib, 10
swmib, 10
tabla, formato, 33
tablas, 33
tipos de acceso
accesible por notificación, 37
no accesible, 37
sólo lectura, 37
tipos de objeto, 27, 31
véase también MIB empresariales, 40
véase también MIB-II
MIB empresariales (enterprise MIB), 40
desarrollar, 40
números asignados a empresas, 39, 40
MIB estándar y privadas, 38
MIB UCD, 335
MIB-I, 254
MIB-II, 9, 42, 253-330, 335, 336
definición, 31
en grupo, salida desde una consulta de tabla *versus* un sistema Linux, 275
grupo de protocolo de Internet, *véase* grupo IP, 277
grupo de reenvío IP
ipForwardDest, 295
ipForwardNextHop, 295
ipForwardNextHopAs, 295
ipForwardNextHopAs, 295
ipForwardPolicy, 295
ipForwardProto, 295
ipForwardTable, 295
ipRouteTable, 295
objetos nuevos, 295
tabla de reenvío IP, 294
vista de ipForward, 294
grupo de traducción de direcciones
atIfIndex, 275, 276
atNetAddress, 275, 277
atPhysAddress, 275, 276
atTable, 275, 276
caché ARP, 275
ifEntry, 276
véase también MIB-II
grupo del sistema
capas funcionales para determinar sysServices, 257
objeto sysUpTime, 419
grupo dot3, 314
dot3CollCount, 319
dot3CollEntry, 319
dot3CollFrequencies, 320
dot3CollTable, 319
dot3ErrorInitError, 320, 321
dot3ErrorLoopbackError, 320
dot3Errors, 320
dot3StateAlignmentErrors, 316
dot3StateCarrierSenseErrors, 318
dot3StateDeferredTransmission, 317
dot3StateEntry, 315
dot3StateEtherChipSet, 319
dot3StateExcessiveCollisions, 317
dot3StateFCSErrors, 316
dot3StateFrameTooLong, 318
dot3StateIndex, 315, 321
dot3StateInternalMacReceiveErrors, 318
dot3StateInternalMacTransmitErrors, 318
dot3StateLateCollisions, 317
dot3StateMultipleCollisionFrames, 316

MIB-II (*continuación*)
dot3StateSingleCollisionFrames, 316
dot3StateSQETestErrors, 317
dot3StateTable, 315
dot3TestLoopBack, 321
dot3Tests, 320
dot3TestsTDR, 320
ethernet-csmacd, 314
iso88023-csmacd, 314
lista de objetos dot3CollTable, 319
lista de objetos dot3Errors, 320
lista de objetos dot3StateTable, 315
lista de objetos dot3Tests, 320
starlan, 314
grupo EGP, 313
grupo de transmisión, *véase* MIB-II, grupo dot3, 314
grupo ICMP, 295
icmpAddrMasks, 298
icmpAddrMasksReps, 299
icmpEchoReps, 298
icmpInDestUnreachs, 296
icmpInEchos, 297
icmpInErrors, 295
icmpInMsgs, 295
icmpInParmProbs, 297
icmpInRedirects, 297
icmpInSrcQuenchs, 297
icmpInTimeExcds, 297
icmpInTimestampReps, 298
icmpOutAddrMaskReps, 302
icmpOutAddrMasks, 302
icmpOutDestUnreachs, 299
icmpOutEchoReps, 301
icmpOutEchos, 301
icmpOutErrors, 299
icmpOutMsgs, 299
icmpOutParmProbs, 300
icmpOutRedirects, 300
icmpOutSrcQuenchs, 300
icmpOutTimeExcds, 300
icmpOutTimestampReps, 301
icmpOutTimestamps, 301
icmpTimestamps, 298
lista de objetos, 295
salida de una consulta al, de un agente Linux, 302
grupo interfaces, 262
consulta ifTable del sistema Linux, salida desde, 263
estado de ifAdminStatus, 267
ifAdminStatus, 267, 270
ifDescr, 265, 269, 284
ifEntry, 268
ifIndex, 262, 268, 284, 285
ifInDiscards, 271
ifInErrors, 272
ifInNUcastPkts, 271
ifInOctets, 271
ifInUcastPkts, 271
ifInUnknownProtos, 272
ifLastChange, 270
ifMtu, 265, 269
ifNumber, 262, 265, 268
ifOperStatus, 267, 270
ifOutDiscards, 273
ifOutErrors, 274
ifOutNUcastPkts, 273
ifOutOctets, 272
ifOutQLen, 274
ifOutUcastPkts, 273
ifPhysAddress, 266, 270
ifSpecific, 267, 274
ifSpeed, 265, 269
ifTable, 262, 268
ifType, 265, 269
objetos ifTable, lista de, 262
RFC1213, 265
tipos de interfaces comunes, 266
grupo IP, 277
ipAddrTable, 283, 284
ipAdEntAddr, 284
ipAdEntBcastAddr, 284
ipAdEntNetMask, 284
ipDefaultTTL, 277
ipForwarding, 277
ipForwDatagrams, 279
ipFragOK, 282
ipFragsCreates, 283
ipFragsFails, 282
ipInaddrErrors, 279
ipInDelivers, 280
ipInDiscards, 280
ipInHdrError, 278
ipInReceives, 278
ipInUnknownProtos, 279

ipNetToMediaIfIndex, 292
ipNetToMediaNetAddress, 293
ipNetToMediaPhysAddress, 293
ipNetToMediaTable, 283, 284
ipNetToMediaType, 293
ipOutDiscards, 280
ipOutNoRoutes, 281
ipOutRequests, 280
ipReasmFails, 282
ipReasmOKs, 282
ipReasmReqds, 281
ipReasmTimeout, 281
ipRouteAge, 288, 291
ipRouteDest, 285, 288, 289
ipRouteEntry, 289
ipRouteIfIndex, 285, 289
ipRouteInfo, 288, 292
ipRouteMask, 288, 291
ipRouteMetric1, 286, 289
ipRouteMetric2, 289
ipRouteMetric3, 290
ipRouteMetric4, 290
ipRouteMetric5, 288, 291
ipRouteNextHop, 286, 290
ipRouteProto, 286, 287, 291
ipRouteTable, 283, 284, 285, 286
ipRouteType, 286, 290
ipRouteType, valores, 286
ipRoutingDiscards, 283
objetos ipRouteTable, lista de, 288
problemas ipRouteTable, 294
salida de ipAddrTable, 285
salida de ipNetToMediaTable, 294
valores ipNetToMediaType, 294
valores ipRouteProto, 287
vista de ipNetToMediaTable, 292
vista de ipRouteTable, 286
grupo SNMP, 321
lista de objetos de la tabla snmp, 323
objetos depreciados (abandonados) debido a SNMPv2, 322
snmpEnableAuthenTraps, 329
snmpInASNPParseErrs, 324
snmpInBadCommunityNames, 323
snmpInBadCommunityUses, 324
snmpInBadValues, 325
snmpInBadVersions, 323
snmpInGenErrs, 325
snmpInGetNexts, 326
snmpInGetRequests, 326
snmpInGetResponses, 327
snmpInNoSuchName, 324
snmpInPkts, 323
snmpInReadOnlys, 325
snmpInSetRequests, 327
snmpInTooBigs, 324
snmpInTotalReqVars, 325
snmpInTotalSetVars, 326
snmpInTraps, 327
snmpOutBadValues, 328
snmpOutGenErrs, 328
snmpOutGetNexts, 329
snmpOutGetRequest, 328
snmpOutGetResponses, 329
snmpOutNoSuchNames, 328
snmpOutPkts, 323
snmpOutSetRequests, 329
snmpOutTooBig, 327
snmpOutTraps, 329
valores válidos, trampas de autenticación, 330
grupo system, 256
resultados de consulta Linux, 261
grupo TCP, 303
lista de objetos, 303
salida tcpConnState, 307
sistema Linux, resultados mediante sondeos a un, 307
tcpActiveOpens, 304
tcpAttemptFails, 305
tcpConnEntry, 307
tcpConnState, 307
tcpConnTable, 306, 307
tcpCurrEstab, 305
tcpEstabResets, 305
tcpInErrors, 306
tcpInSegs, 305
tcpMaxConn, 304
tcpOutRsts, 307
tcpOutSegs, 306
tcpPassiveOpens, 304
tcpRetransSegs, 306
tcpRtoAlgorithm, 303
tcpRtoMax, 304
tcpRtoMin, 304
valores tcpConnState, 308

MIB-II (*continuación*)
valores tcpRtoAlgorithm, 303
vista tcpConnTable, 307
grupo UDP, 309
agente SNMP de Linux, resultados de consultas al, 311
lista de objetos, 309
udpEntry, 310
udpInDatagrams, 309
udpInErrors, 309
udpLocalAddress, 310
udpLocalPort, 311
udpNoPorts, 309
udpOutDatagrams, 310
udpTable, 309, 310
grupos, lista de, 254
MIB Ethernet, 314
MIB Ethernet, métodos de acceso, 314
objetos, 256
determinar valores de dispositivos
sysServices, 256
sysContact, 258, 262
sysLocation, 259, 262
sysName, 259, 262
sysObject, 262
sysObjectID, 258
sysOR prefixed, 256
sysORDescr, 261
sysOREntry, 260
sysORID, 260
sysORIndex, 260
sysORLastChange, 259
sysORTable, 260
sysORUpTime, 261
sysServices, 256, 259, 262
sysUpTime, 256, 257, 258
vista
de ipAddrTable, 284
de la tabla dot3StateTable, 315
vista del grupo, 255, 275
de transmisión dot3, 314
icmp, 296
interfaces, 263
ip, 278
snmp, 322
system, 256
tcp, 303
udp, 311
vista general de, 254
minuciosidad de la conectividad, 150
modelo
de control de acceso basado en la vista (VACM), 348
de objeto de entorno de red GNU, 546
de seguridad del usuario (USM), 335, 350
modelo OSI, 62
capa
de aplicación, 62
de enlace de datos, 63
de presentación, 62
de red, 63, 84
de transporte, 63
física, 63
modelo TCP/IP, 63
capa
de acceso a la red, 65
host a *host*, 63, 72
Internet, 65
capas
asignadas para capas OSI, 63
de proceso, 64
lista de, 63
servicios de la capa de proceso, 65
modo
de acceso, 31, 305
terminal, uso del, 543
módulo(s)
del sistema, 549
tr, 549
MSL, 81
MTU, 124
multi-homing, 115

N

netmask, 125
netstat, 233
bandera
G, 142
H, 142
U, 142
campo
destination (destino), 141
Flags (banderas), 142
Flg, 139

gateway (puerta de enlace), 141
Genmask, 142
Iface, 143
irtt, 143
Metric, 143
MMS, 142
PID, 138
Ref, 143
RX-DRP, 139, 140
RX-ERR, 139
RX-OK (recibido), 139
RX-OVR, 140
TX-DRP, 139, 140
TX-ERR, 139, 140
TX-OK (transmitido), 139
TX-OVR, 140
Use, 143
Window (ventana), 142
campos
nombre del programa (Program name), 138
State, 132
estadísticas de protocolo, presentación de las, 144
estado(s)
ESTABLISHED, 132
LISTEN, 135
TCP mostrados con, 133
identificación de riesgos de seguridad, 136
información
de enrutado, presentación de la, 141
de interfaz, presentación de, 138
de multidifusión, presentación de la, 144
interfaz lógica, 140
lista de funciones, 129
objetos MIB que se asignan a la información netstat, 140
opción
--statistics, 144
-a, 133
-c, 146
-e, 143
-g, 144
-p, 137
-r, 141
-t, 132, 133, 146
-u, 133
opciones
adicionales de línea de comandos, 146
del comando, 130
palabra clave
--interfaces, 141
--raw, 145
--tcp, 145
--udp, 145
--unix, 136
raw, 144
palabras clave, 129
resultados de netstat --help, 131
sesiones de red activas, presentación de las, 132
tablas de enrutamiento, 141
NFS, 70
NIS, 70
niveles de ejecución, 548
determinar el nivel actual, 548
modo multiusuario, 548
nivel de ejecución
2, 548
3, 548
5, 548
nmap
descripción, 224
explorar una red para un puerto en particular, 234
instalación de, 642
listado
de salida, 240
estándar de red, 240
obtención de acceso a la red, 225
opción
-O, 227
-sT, 233
-sU, 231, 233
-v, 236
opciones
adicionales, 229
de exploración, 227
rangos de exploración IP, 235
reporte del sistema
AIX, 242
Cisco Router, 245
HP-UX, 244
Solaris, 243
Windows NT, 245
técnicas de exploración, 228

NNTP, 72
no confiable, 85
nodos, 25
nombre
  de comunidad, 40
  del objeto, 31, 304
notación abstracta de sintaxis (ASN.1)
  véase ASN.1
notación decimal punteada, 126
notación punteada, 25
  véase también MIB, notación punteada, 25
ntop, uso de las opciones de comando, 540
null scan (exploración nula), 229
número empresarial de Sun, 388

O

objetos específicos Sun enterprise, 336
objetos MIB
  alteración de, 441
  traducir a un formato más amigable y legible, 450
  véase también herramienta snmpset, 440
OID, 31, 304
operación
  de protocolo SNMP, 44
  del SNMP, 23
operadores
  aritméticos, 173, 174
  booleanos, 173, 175
OSI, capa(s)
  de aplicación, 62
  de enlace de datos, 63
  de presentación, 62
  de red, 63
  de sesión, 63
  de transporte, 63
  física, 63
  lista de, 62

P

panel de control, 124, 546
  véase panel de control Linux, 546
panel de control Linux, 545-580
  acceso al panel de control, 546, 547
  alias de interfaz, 564
  botón
    Done (hecho), 554
    Edit (editar), 554
    Remove (remover), 553
    Save (guardar), 554
  campo netmask, 556
  configuración
    de interfaces de red, 554
    de la red, 549
    del sistema, 549
  configurador de la red, 547, 549
    botón Hosts (anfitriones), 551
    botón Interfaces, 554
    botón Names (nombres), 550
  configurador del sistema
    categoría Control, 576
    categoría cuentas de usuario, 573
    categoría de servicios misceláneos, 576
    funciones de categoría Control, 578
    funciones de la categoría cuenta de usuario, 574
    funciones de modo de arranque, 577
    funciones de sección de trabajo en red, lista de, 569
    funciones de servicios misceláneos, 576
    sección de trabajo en red, 569
    sección sistemas de archivo (File systems), 576
    sistemas de archivos, 576
    subcategorías de cuentas de usuario, 573
    uso del, 568
  dirección de red para la máquina, crear otra, 564
  editor de nivel de ejecución, 548
  entradas de *host*, 553
  fecha y la hora, cambio de la, 548
  funciones, lista de, 546
  gnome-linuxconf, 568
  help (ayuda), 549
  icono del configurador de red, 550
  impresoras, configuración y control de, 548
  interfaz externa, agregar una nueva, 555
  mantener la base de datos de *host*, como, 551
  orientación de visualización de la ventana principal, 547
  parámetros de configuración DNS, 550
  ppp, 558

protocolo
de arranque (BOOTP), 557
de configuración dinámica del *host* (DHCP), 557
subcategorías de configurador de sistemas, 569
uso de los módulos del kernel, 549
ventana
de alias de interfaz, 564
de clon, 564
de configuración del sistema, 568
de diálogo de *host* agregado, 553
de diálogo Save, 557
de interfaz Ethernet, 556
de selección de la interfaz, 555
del configurador de red, 550
verificación de error, 553
vista
de *hosts*, configurador de red, 552
de la ventana principal del, 548
vista interfaces
Activate (activar), 554
Alias (alias), 554
Clone (clon), 554
Deactivate (desactivar), 554
paquete de red Scotty, 581
paquete internet groper
véase ping, 147
paquetes, 34
de entrada, 34
de red, datos estadísticos acerca de los, 511
de salida, 34
de software snmpconf y CMU, instalación de los, 656
de unidifusión, 34
diferentes de unidifusión, 34
par
de identificación de la red, 86
de nodos de identificación, 86
ping
descripción, 147
desempeño de la red, determinación del, 152
disponibilidad del sistema, determinación del, 148
fallas de red, aislamiento, 150
límites, 149
número de secuencia, 154
opción
-c, 149
-f, 156
-i, 155
-n, 156
-R, 156
-s, 153
opciones
de comando, 151
varias, 156
rendimiento de la red, evaluación del, 153
sesión de parada, 149
solicitudes, controlando el número de, 149
tráfico de red, simulando con la opción de inundación, 157
ping sweep, 229
PPP, 558
(des)activar conexiones, 560
control de flujo, 559
inicio del sistema de arranque, 561
opción de depuración, 560
opciones
de reinicio de conexión, 562
especiales, 559
panel
Communication (comunicación), 560
de configuración PAP, 563
de Networking (red), 561
PAP, 563
parámetros de configuración, 558
protocolo PAP, 558
puerto de hardware, 560
reiniciar cuando la conexión falla, 562
ventana
de diálogo de interfaz, 558
de marbete comunicación, 561
de personalización, 559
preferencias de impresión ethereal, 210
primitivas
aritméticas, 169
de direcciones, 169
de protocolo, 169
programación de secuencias de comandos, 217
protocolo común de información para la administración (CMIP), 14

protocolo de administración de red simple (SNMP)
véase SNMP
protocolo de configuración dinámica de *host* (DHCP)
véase DHCP, 69
protocolo de control de transmisión (TCP)
véase TCP/IP
protocolo de datagrama del usuario (UDP)
véase MIB-II, grupo UDP, 309
protocolo de descubrimiento de enrutador (RDP), 70
protocolo de información de enrutado (RIP), 70
protocolo de Internet (IP)
véase IP, 84
véase también TCP/IP
protocolo de mensaje de control de Internet (ICMP)
véase ICMP
protocolo de puerta de enlace externa (EGP)
véase MIB-II, grupo EGP, 313
protocolo de resolución de direcciones (ARP)
véase ARP, 69
protocolo de transferencia de archivos (FTP)
véase FTP, 60, 67, 69
protocolo de transferencia de hipertexto (Hyper Text Transfer Protocol), 69
protocolo de transferencia de noticias en red (NNTP) *véase* NNTP
protocolo invertido de resolución de dirección (RARP)
véase RARP, 69
protocolo no orientado a la conexión, 55
protocolo simple de transferencia de correo (SMTP), 71
protocolo trivial de transferencia de archivos (TFTP), 71
protocolo UDP, 71
protocolos
CMIP, 14
SNMP, 14
TCP/IP, 14
proveedor de servicios de Internet (ISP), 87
prueba de ciclo de retorno del hardware, 320
pseudointerfaces, 34, 36, 127
véase también ifconfig, interfaz lógica, 127
PSH, 83
puerta de enlace predeterminada, 567
puerto(s), 67, 75
inhabilitar, 67
proceso asociado con, 313
UDP, 228
y transformación entre los procesos UNIX, 538

R

RADIO, 71
rama enterprises, 469
RARP, 69
formato de mensaje, 102
véase también ARP, 100
RDISC, 70
red
administración del rendimiento, 535
comunidades de usuarios, 40
cuellos de botella, 2
descubrimiento de la, 18
estadísticas de tráfico, 522
funcionando, 149
inundaciones de, 157
inventario, *véase también* herramienta snmpstatus, 446
límite de seguridad, 40
mapas de topología de la, 581
medición del rendimiento, 157
nombre de comunidad, 40
particionar dispositivos para reducir la sobrecarga de la, 535
problemas
de conectividad de, 295
de la capa de, 295
de rendimiento, 474
potenciales, determinar, 440
procesamiento avanzado de alarmas, 20
red de oficina, ejemplo de una, 475
rendimiento
de la, 440
general de la, 524

reportes de tráfico, 534
seguimiento
de errores del trabajo en, 429
del rendimiento, 17
términos sobre la administración de, 659
tráfico de unidifusión, 526
velocidades, 484
de tipos de interfaz, 485
zonas, 40
véase también herramienta ntop, categoría de tráfico IP, 533
red de área amplia (WAN), 3
red virtual de área local (VLAN), 3, 20
redes de área de almacenamiento, 3
redes virtuales privadas (VPN), 3, 20
reflectometría en el dominio del tiempo (TDR), 320
rendimiento de la interfaz, 420
rendimiento de la red, 474
establecer una línea base, 524
reportes remoto y local, 534
resultados del reporte, 242, 243, 244, 245
Reverse Indent (sangría contraria), 228
REXEC, 70
RFC
1156, 254
1213, 254
1227, 58
1441-1452, 42
1643, 314
1901-1908, 42
1901-1908, 42
1909-1910, 42
2222, 42
2271-2275, 43
proceso, 39
RIP, 70
RLOGIN, 70
RPC, 70, 605
RSH, 70
RST, 83
rstatd, 609
RTT, 615
RWALL, 71
RWHO, 71

S

SAN, 3
secuencia de comandos df2mrtg, 502
opciones de línea de comandos de la, 504
segmento, 64
ACK, 78, 80, 83
final (FIN), 79, 80, 229
segmento de mensaje de sincronización
véase SYN
seguimiento de la ruta, 420
servicio de conversación en tiempo real en Internet (IRC)
véase IRC, 72
servicio de dominio de nombre (Domain Name Service) (DNS)
véase DNS
servicio de ejecución remota (REXEC), 70
servicio de información de red (NIS),
véase NIS, 70
servicio de inventario, 235
servicio de shell remoto (RSH), 70
servicio remoto de inicio de sesión (RLOGIN), 70
servicios centrales, 15
servicios de administración de información (MIS), 596
servicios del sistema, 67
sesión dúplex, 76
Set Parameter, 610
SetRequest, 47
simulación de tráfico de red, 157
síndrome de conjunto de no retorno, 443
sistema de administración de la red, 1, 2
administración del rendimiento, 2
sistema de administración de red, 1, 2
administración
de diagnósticos, 1, 3
de la configuración, 1, 3
administradores, relación de agentes con, 4
seguimiento del rendimiento, 1
sistema de archivos de red (NFS), *véase* NFS, 70
sistema *multi-homed*, 186

SMTP, 71
SMUX, 57
SNMP, 6, 11, 14, 21-58
  arquitectura y seguridad, 41
  autenticación, 40
  beneficios, 22
  códigos de respuesta, 52
    genError, 53
    InconsistentName, 54
    InconsistentValue, 54
    noAccess, 53
    NoCreation, 54
    NoSuchName, 52
    NotWritable, 53
    Read-only, 53
    resourceUnavailable, 54
    tooBig, 52
    WrongEncoding, 53
    WrongLength, 53
    WrongType, 53
    WrongValue, 54
  comunidades, 40
  descripción, 22
  encriptación, falta de facilidad de, 41
  equipo no tradicional de control, 22
  formato de mensaje, 44
    nombre de comunidad, 44
    unidades de datos de protocolo, 45
    versión, 44
  funciones, 23
    get, 23
    set, 23
    trap, 23, 24, 48
  huella, 22
  información de control de acceso a los objetos MIB, 37
  mensaje, transmisión de un, 54
  operación de protocolo, 44
  preguntas y respuestas, 45
  protocolo no orientado a la conexión, 55
  seguridad, 40
SNMPv1, 41, 42
SNMPv2, 42
SNMPv2c, 42
SNMPv2usec, 42
SNMPv3, 43
  trampas predefinidas de, 50
  versiones de, 41
  vista detallada, 331
SNMP, vista general de las versiones de, 41
Snmptable, 466
SNMPv1, 41, 42, 52, 335, 336
  desventajas, 41
SNMPv2, 43, 334, 335
  vista general de, 43
SNMPv2 Classic, 42
SNMPv2/v3, 53
SNMPv2c, 43
SNMPv2usec, 42
SNMPv3, 43, 334, 335
  instalación, 43
  vista general, 43
snmpwalk, 433
snoop, 207
software Chesapeake, instalación del, 648
solicitudes de comentarios (RFC), 9
  véase también RFC
Solstice, 331
sondas, 17
  de red, 17
subred(es), 88
  calculador de dirección, 91
  cálculo de la, 88, 89
  direccionamiento de la, 88
  pasos para crear, 90
SUNWascom, 651
SUNWmibii, 651
SUNWsadmi, 651
SUNWsasnm, 651
SYN, 78, 80, 83

T

tabla(s), 33
  de carga del sistema UCD, 380
  de conexión TCP, 447
  de disco UCD, 376
  del agente extensible UCD, 374
  tcpConnTable, 466
  UDP, ejemplo de la, 434
TALK, 71
tamaño máximo de segmento (MSS), 142
tamaños de marcos de enlace de datos, 198
tasa de baudios, 560
TCP
  cerrar conexiones, 78
  conexiones

cerrar, 78
conteo de mensaje para, 79
establecer, 78
estados, 80
conocimiento de mensajes, 77
formato de secuencia, 81
bits de código, 82
campo de ventana, 83
capa Internet, 84
datos, 82
hlen, 82
MTU, 84
número de reconocimiento, 82
número de secuencia, 82
opciones, 82
opciones TCP, 84
puerto destino, 82
puerto fuente, 82
puntero urgente, 82, 84
suma de verificación, 82
ventana, 82
información operacional y estadística acerca de, 303
segmentos, ajustar dinámicamente el tamaño de los, 84
tipos de segmento, 83
TCP connect, 228
TCP/IP, 14, 59-107
arquitectura, 63
conexiones, 75
virtuales, 73
conjunto de protocolos, 60
datos en secuencia, 73
descripción, 60
dirección
de destino IP, 77
fuente IP, 77
herramientas de usuario final, 66
huellas digitales, 227
independencia del sistema operativo de, 60
interfaz de abstracción de flujo, 74
marco, 65
para administración local, 60
puertos, 75
destino, 77
fuente, 77
RFC, 60
segmento, 64
orden de llegada de, 73
seguridad, 67
servicios, 65
adicionales, 71
centrales, 67
del sistema, 67
y tecnologías de red, 61
zócalos, 75
tcpdump
acceso para usuarios capacitados a, 159
diagrama del filtro de paquete, 169
filtros de paquete
calificadores, 176
calificadores dst, 176
calificadores ether, 176
calificadores src, 176
diversas primitivas, 175
filtros compuestos, construir, 173
operadores, 173
operadores, lista de, 173
para calificadores, 176
primitiva broadcast (difusión), 171
primitiva etheraddr, 171
primitiva ethertype, 172
primitiva ip, 171
primitiva multicast, 172
primitivas de dirección, 169
primitivas de dirección, lista de, 170
primitivas de protocolo, 172
primitivas de protocolo, lista de, 173
primitivas gateway, 171
primitivas greater, 176
primitivas *host*, 170, 176
primitivas ipaddr, 171
primitivas less, 175
modo de captura, 158
no promiscuo, 159
no promiscuo, 158
opción
-c, 161, 162
-e, 165
-i, 161
-p, 163
-q, 159
-t, 162
-v, 165
-w, 163
-x, 166
etheraddr, 171
ipaddr, 171

opciones
de modo de operación, 161
de presentación, 165
palabras clave
gateway, 171
host, 171, 176
promiscuo, 158
tamaño de paquete, determinar, 166
usar tcpslice para cortar o unir archivos tcpdump, 164
uso de filtros de paquetes, 168
tcpslice, 164
TCPSYN, 228
tecnología de redes, 3
mejoras y avances, 3
telnet, 66, 71
rlogin y, 608
términos sobre red, 659
TFTP, 71
tiempo de ida y vuelta (RTT)
véase traceroute, RTT, 181
tiempo de ida y vuelta inicial (*initial round-trip time*, IRTT), 143
tiempo de vida (TTL)
véase traceroute, campo TTL, 177
tipos de datos de aplicación
Counter, 29
DisplayString, 30
Gauge, 30
NetworkAddress, 30
Opaque, 30
TimeTicks, 30
tipos de datos universales, 28
counter32, 29
integer, 28
IpAddress, 30
null, 28
object identifier, 29
octect string, 28
tipos de objeto, 27, 31, 305
tkined, 581
adición de nuevo menú al menú principal, 605
administración de la red, 604
lista de tareas del menú Tools, 605
comando, 582
descubrimiento de la red, 584
diálogo de Selection by type, 630
funciones, lista de, 582
invocar, 582
en un indicador de comandos shell, 582
menú
Edit, 630
File, 628
Help, 628
Icon, 633
Open, 629
Options, 633
Select, 629
SNMP, 623
Structure, 591, 632
Tools, 604
menú IP-Discover, 620
Discover IP Network, 621
Discover Route, 621
IP World Menu, 623
lista de consulta de servidor RPC, 622
Resolve Email, 623
Set Parameter, 623
Show RPC/TCP Server, 622
menú IP-Layout, 618
delete IP-Layout, 620
Group Network, 619
Layout Network, 619
Layout Nodes & Groups, 619
Set Address, 619
Set Icon, 619
Set Name, 619
Show Parameter, 620
Unique Name, 619
menú IP-Monitoring, 611
Check Reachability, 611
CPU, 615
Delete IP-Monitor, 618
Disk, 615
dispositivo de notificación de falla, 612
Ethernet Load/Active Users, 616
Interface Activity, 615
Modify Monitor Job, 616
Monitor Job Info/Modify Monitor Job, 616
Round-Trip Time, 614
Set Parameter, 618
submenús, 605
System Load, 615

ventana de la tarea de seguimiento Modify, 617
vista de rendimiento de monet, 615
vista de rendimiento especifico, 615
menú SNMP-Host, 626
ventana de opciones de parámetros SNMP, 625
menú SNMP-Monitor, 626
gráfico reducido, 627
Interface Load, 627
Monitor Variable, 626
TCP Service User, 627
ventana de diálogo de seguimiento, 627
menú SNMP-Trouble, 623
ICMP y SNMP Statistics, 625
Interfaces, IP, TCP y UDP, 625
Set SNMP parameter, 625
SNMP Devices, 623
System Information, 624
ventana de presentación de la información del sistema, 624
Walk MIB Tree, 625
panel de herramientas
herramienta de cambio de tamaño, 589
herramienta de conexión, 591
herramienta de dispositivo, 590
herramienta de grupo, 591
herramienta de red, 590
herramienta de referencia, 592
herramienta de selección, 589
herramienta de texto, 590
lista de herramientas, 588
uso del, 588
referencia de menús, 627
reglas de proceso de descubrimiento IP, 620
selección de tipo, 630, 631
submenú IP-Trouble, 605
Daytime, 608
Delete IP-Trouble, 611
DNS Info, 608
Ether Status, 610
Finger, 608
Help IP-Trouble, 611
Host status, 609
listado de los servicios RPC, 609
Multi Ping, 607
Netmask, 607
NFS Exports, 609
NFS Mounts, 609
NTP Info, 608
Ping, 607
servicios TCP/RPC, 609
Set Parameter, 610
telnet y rlogin, 608
traceroute, 608
ventana de diálogo de intervalo, 610
ventana de diálogo Multi Ping, 607
ventana de parámetro IP-Trouble, 611
ventana de resultado Ping, 608
ventana de resultado remoto, 610
Whois, 608
ventana
de estado de descubrimiento, 585
de solicitud de descubrimiento de la red, 584
principal tkined, 583
principal, submenús de la, 628
vista
de menú IP-Discover, 621
de submenú IP-Trouble, 606
del panel de herramientas, 589
general de, 582
vista del menú
Edit, 631
File, 629
Icon, 633
Options, 633
Select, 630
Structure, 632
vistas de topología, 583
agregar una imagen de fondo, 600
construcción de vistas, 594
construcción de vistas, enfoques para la, 594
construir una vista LAN, 596
construir una vista WAN, 601
ejemplo de la topología física, 595
ejemplo de la topología lógica, 595
iconos de dispositivo, lista de, 587
iconos personalizados, 586
mover un icono, 585
reemplazar iconos, 586
uso de las imágenes de fondo, 604
vista de grupo expandido, 592
vista de grupos de objetos, 592
vista de topología completada, 587

tkined (*continuación*)
vista de topología física, 594
vista de topología WAN, 603
vista LAN, 596
vista LAN completada, 602
vista lógica de red, 594
vista WAN, 601
vistas, definición, 583
TokenRing, tamaño del mensaje
estándar, 124
traceroute
campo TTL, 177
características operacionales, cambio de las, 185
códigos de presentación de, 182
ejemplo(s)
de dos saltos, 181
de un salto, 180
RTT, 181
lectura de la salida del, 182
opción
-I, 187
-i, 186
-m, 186
-n, 188
-p, 187
-v, 187
opciones
de presentación, 187
operacionales, 185
TTL, 186
tráfico
de red, 139, 194, 514
protocolo de la capa de transporte, 539
red de captura, 460
tráfico de captura, 460
de unidifusión, 526
IP, medición del, 277
trampa (Trap), 49, 419
campos
dirección del agente, 49
empresa, 49
pdu, 49
Time-stamp (sello de tiempo), 50
Trap especifico, 49
Trap genérico, 49
vínculo variable, 50
formato de mensaje, 49
tipos
AuthenticationFailure, 51
ColdStart, 50
EgpNeighborLoss, 51
EnterpriseSpecific, 51
LinkDown, 50
LinkUp, 50
WarmStart, 50
trampas de fallas-autenticación, grupo SNMP, 321
véase también MIB-II
transmisión de un mensaje SNMP, 54

U

unidad máxima de transmisión (MTU)
véase TCP, formato de secuencia, 84
unidades de datos de protocolo (PDU), 42
UNIX, 6
plataforma más popular, 15
URG, 83, 84
URL, 106
uso de la red, 537
utilidad mbiisa
véase agente SNMP de Sun, 396
utilidad pkgadd, 336
utilización del espacio del disco, 475

V

VACM, 348
valor, 37
ventana
de Captura/Playback ethereal, 199
de filtro de paquete TCP, 203
de filtro ethereal, 204
de opciones de presentación del campo time ethereal, 211
de preferencias de captura ethereal, 199
de preferencias ethereal, 209
de resumen del paquete ethereal, 215
vida máxima del segmento (MSL)
véase MSL, 81
vinculación (*binding*)
dirección estándar usada para vincularse a un puerto local, 312
vista general de SNMPv1, 41

W

WAIS, 72
WHOIS, 72

X

Xmax Tree, 229
xtraceroute
  adicionando *hosts*, 248
  base de datos personal del usuario para, 251
  campo
    Accuracy, 250
    Minimum Distance, 250
    Ping time, 250
  descripción, 246
  información
    de salto de destino, 249
    del rendimiento, 249
  instalación de, 642

Z

zócalos, 75, 76, 77
  asignación en los sistemas UNIX, 77

Licencia sobre software e información

El software y la información que contiene este(os) CD-ROM(s) (los cuales son llamados conjuntamente el "Producto") son de propiedad de The McGraw-Hill Companies, Inc. ("McGraw-Hill") y están protegidos tanto por la ley de derechos de autor de los Estados Unidos, como por el tratado internacional sobre derechos de autor. Usted debe usar este producto como si se tratase de un libro y se espera que lo copie en una computadora para poder utilizarlo; puede hacer archivos de copia del producto con el único propósito de tener copias de seguridad de su software y para proteger su inversión de posibles pérdidas

Cuando se estipula "como si se tratase de un libro", McGraw Hill quiere decir, por ejemplo, que una cantidad indeterminada de personas puede usar el producto y este puede trasladarse libremente de una computadora a otra, siempre y cuando no exista la posibilidad de que el producto ,o cualquier parte de el) se use en un sitio o en una computadora, mientras se usa en otra. "Como si se tratase de un libro" significa también que no pueden leerlo dos personas distintas a la vez en dos sitios diferentes, ni tampoco puede usarse en dos sitios diferentes al mismo tiempo (a menos, por supuesto, que se violen los derechos de McGraw Hill .

McGraw-Hill se reserva el derecho de alterar o modificar el contenido del producto en cualquier momento.

Este acuerdo es efectivo hasta que se dé por terminado. El acuerdo se terminará automáticamente sin notificación alguna si usted falla en el cumplimiento de cualquier clausula que haga parte de él. En caso de que este acuerdo se dé por terminado por incumplimiento de parte suya, usted deberá destruir o eliminar todas las copias del producto instaladas en cualquier sistema de computadoras, o que se hayan elaborado con el propósito de mantener copias de seguridad y deberá borrar el producto de todos sus disposit.vos de almacenamiento de datos.

Garantía limitada

McGraw Hill garantiza que el(los) CD-ROM(s) adjunto(s) está(n) libres de defectos en sus materiales y en su elaboración durante un periodo de sesenta días después de la fecha de compra Si McGraw-Hill recibe cualquier notificación, por escrito, dentro del periodo de garantía acerca de defectos en los materiales o en la elaboración, McGraw-Hill determinará si dicha notificación es correcta y remplazará el,los) CD-ROM(s) defectuoso(s). Envíe su solicitud a:

Servicio al cliente
McGraw-Hill Interamericana S.A.
Av. Américas #46-41 Tel. (0571) 3682700-3377800, Ext. 106 Fax (0571) 3687484
Bogotá, Colombia

La responsabilidad y solucion total del incumplimiento de esta Garantía Limitada se limitará al remplazo del(los) CD-ROM(s) defectuoso(s) y no incluirá ni extenderá ningún reclamo o derecho a cubrir cualesquiera otros daños, incluyendo, pero sin limitarse a pérdidas de utilidades, datos, o uso del software, o daños especiales, incidentales o derivados, u otros reclamos similares, aunque McGraw-Hill haya sido específicamente notificada acerca de la posibilidad de dichos danos En ningún caso la responsabilidad de McGraw Hill por cualquier daño causado a usted o a alguna otra persona podrá exceder el precio más bajo de la lista de precios sugerida, o el precio realmente pagado por la licencia para usar el producto, sin importar cuál sea la forma del reclamo.

The McGraw Hill Companies, Inc. específicamente desconoce cualquier otra garantía, expresa o implícita, incluyendo, pero sin limitarse, cualquier garantía de comercialización o disposición para un propósito particular. De manera específica, McGraw-Hill no hace ninguna representación o garantía de que el producto sea apto para un propósito particular y que cualquier garantía implícita de comercialización está limitada a los sesenta días de duración de la Garantía Limitada que cubre el(los) disquete,s físico(s) solamente (y no el software o la información) y que es, en caso contrario, expresa y específicamente rechazada

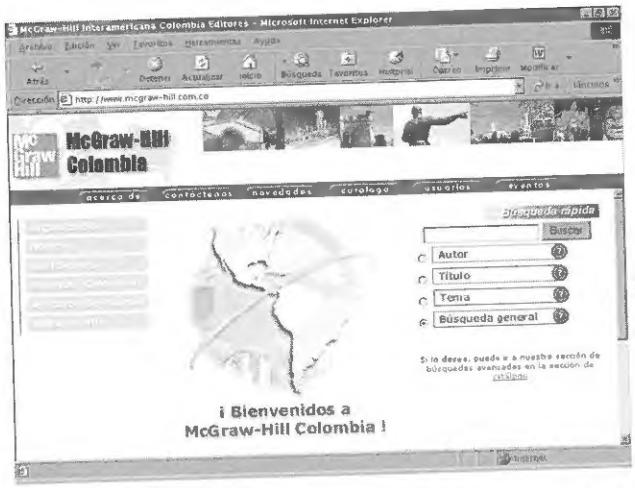
Esta Garantía Limitada le confiere a usted derechos legales específicos; aunque puede tener otros que pueden variar de país a país Algunos países no permiten la exclusión de daños incidentales o consecuenciales, o el límite de duración de una garantía implícita, por tanto, algunos de los mencionados pueden no ser aplicables a su caso.

Este acuerdo constituye el acuerdo total entre las partes en lo relacionado con el uso del producto. Los términos de cualquier orden de compra no tendrán ningún efecto sobre los términos de este acuerdo. El hecho de que McGraw Hill falle en insistir sobre el estricto cumplimiento de este acuerdo no constituye una renuncia a ninguno de los derechos contemplados en él. Este acuerdo deberá elaborarse y gobernarse de acuerdo con las leyes de cada país. Si se determina que cualquier cláusula contenida en él es contraria a la ley, dicha cláusula tendrá el máximo efecto permisible y las restantes cláusulas permanecerán vigentes y efectivas.

Otros títulos de referencia

Gallego	Unix/Linux, Iniciación y Referencia	8448121015
Shan	Linux, Manual de administración	8448128923
López	Linux, Guía de instalación y administración	8448128915
Martínez	Linux, La referencia visual	958410182X
Petersen	Linux, Manual de referencia	8448109694
Petersen	Fundamentos de programación en Linux	

CONÉCTESE A
www.mcgraw-hill.com.co
Y contáctenos



OFICINAS IBEROAMERICANAS

ARGENTINA
McGraw-Hill Interamericana, Ltda.
Suipacha 745- 8.°
(1008) Buenos Aires
Tel. (541) 322 05 70. Fax: (541) 322-1538

BRASIL
McGraw-Hill do Brasil
Rua da Assenbléia, 10/2319
20011-000 Río de Janeiro
Telefax: (5521) 531-2318

CARIBE
McGraw-Hill Interamericana del Caribe
Avenida Muñoz Rivera, 1121
Río Piedras
Puerto Rico 00928
Tel. (809) 751-3451 / 751-2451
Fax: (809) 764-1890

CHILE, PARAGUAY Y URUGUAY
McGraw-Hill Interamericana de Chile, Ltda.
Pocuro 2151 Providencia
Santiago (Chile)
Tel. (562) 373-3000 Fax: (562) 221-0452

COLOMBIA, ECUADOR, PERÚ y BOLIVIA
McGraw-Hill Interamericana, S.A.
Apartado 81078
Avenida de las Américas # 46-41
Bogotá (Colombia)
Tel: (571) 368-2700 / 337-7800. Fax: (571) 368-7484

ESPAÑA
McGraw-Hill Interamericana de España, S. A. U.
Edificio Valrealty, Planta 1ª.
Basauri, 17
28023 Aravaca (Madrid)
Tel: (341) 372-8193. Fax: (341) 372-8513
E-mail: profesional@mcgraw-hill.es

GUATEMALA
McGraw-Hill Interamericana Editores, S.A.
11 Calle 0-65, Zona 10
Edificio Vizcaya, 3er. nivel
Guatemala, Guatemala
Tel. (502) 332-8079 al 332-8084
Fax: (502) 332-8114

MEXICO Y CENTROAMÉRICA
McGraw-Hill Interamericana Editores S. A. de C.V.
Atlacomulco 499-501
Fracc. Ind. San Andrés Atoto
53500 Naucalpan de Juárez
Estado de México
Tel. (525) 628-5353 Fax: (525) 628-5302
Cedro, 512-Col. Atlampa
06460, México D. F.
Tel. (525) 171-1515 Fax: (525) 117-1589
Centro Telemarketing
Tel. (525) 628-5352 / 628-5327 Fax: (525) 628-8360
Lada. sin costo 91 8834 540

PANAMÁ
McGraw-Hill Interamericana de Panamá, S.A.
Edificio Banco de Boston, 6° piso Oficina 602
Calle Elvira Méndez
Panamá, Rep. De Panamá
Tel. (507) 269-0111. Fax: (507) 269-2057

PORTUGAL
Editora McGraw-Hill de Portugal, Ltda.
Estrada de Alfragide, lote 107, bloco A-1 Alfragide
2720 Amadora (Portugal)
Tel. (35611) 472-8500 Fax: (3511) 471-8981

USA
McGraw-Hill Inc.
28th floor 1221 Avenue of the Americas
New York 10020
Tel. (1212) 512-2691. Fax: (1212) 512-2186

VENEZUELA
McGraw-Hill de Venezuela, S.A.
Apartado Postal 50785
Caracas 1050
Edificio torre solano
Avenida francisco solano, sabana grande
Parroquia el recreo
Tel: (582) 761-8181/ 762-5562. Fax: (582) 762-8224